Wavelet Modulation and Transmission Through Fading Channels

Todd K. Moon and Chet Lo
Electrical and Computer Engineering Department
Utah State University

Abstract

In separate parts, this report addresses: (1) the use of multiscale modulation to obtain bandwidth-efficient modulation, and the performance of such modulation in fading channels; (2) The analysis of transmission through fading channels, and approaches to the design of transmitter and receiver waveforms which will perform optimally in fading channels. In the first case, it is shown that using wavelet and/or scaling waveforms can lead to modulation which is comparable to, or better than, many conventional schemes, including GMSK. Performance in fading channels is similar to that of other methods. The performance of multiscale signaling is also addressed using the matched-filter bound method. In the second part, the signal design problem is formulated as an optimization problem and an analysis approach outlined which leads to maximization of SNR in fading channels. Inclusion of constraints for good spectral localization is also possible.

1 Introduction

In this report, results from two related directions of research are presented. First, results related to wavelet modulation are discussed. Second, results related to waveform design for communication over fading channels are presented.

Wavelet modulation employs as its baseband basis waveforms the shift- and scale-orthogonal scaling functions and wavelets, typified by Daubechies wavelets (see, e.g., [1]). Because of the orthogonal properties, it is possible to employ multiscale modulation, sending information on multiple time scales.

One of the potential benefits of wavelet modulation is that, as basis functions with longer support are used (resulting in multiple time-overlapped waveforms), the smoothness of these waveforms leads to better spectral localization, as measured by the fraction of out-of-band power (FOOBP), than many other popular signal formats, including GMSK.

As one of the research hypothesis, it was conjectured that transmitting on multiple scales and over long time frames might yield a performance benefit in channels experiencing fast fading. The conjecture arose with the idea that a short fading interval might affect only a small fraction of the overall waveform. A significant part of the research project was devoted to exploring this question. The findings are that in some cases, there may be a modest benefit in a fading channel, in comparison to conventional (flat-pulse) signaling. However, the improvements seemed to apply to such a narrow set of specific cases, and in any event be sufficiently modest, that it seemed difficult justify the use of these basis functions on that criterion alone.

This leads to the question of whether there might be other waveforms which could be specifically designed to be effective in fading channels and which might be, furthermore, constrained to have a sufficiently high degree of spectral localization that they could be used in practical channels. This therefore, forms the turning point for the second part of the research, into waveform design for fading channels. As of the time of this report, approaches to the problem have been formulated, as will be discussed, but the performance of these methods compared with other channels remains to be accomplished.

The organization of this report is as follows. Sections 2 through 3.2 present the basic notation, spectral efficiency, and analysis for communication over fading channels using differential detection. In section 4, the matched filter bound is employed to analyze multiscale transmission over several fading profiles. It is found that the performance is somewhat better than raised cosine waveforms in many cases, but again

REPORT DOCUMENTATION PAGE

AFRL-SR-BL-TR-00-

0238

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
5-15-00	Final	

4. TITLE AND SUBTITLE

Wavelet modulation for bandwidth efficient and fading resistant modulation.

5a. CONTRACT NUMBER

5b. GRANT NUMBER

F 49 6 20-99-1-0024

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Todd Moon
Chet Lo

5d. PROJECT NUMBER

5e. TASK NUMBER

2304/KX

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Utah State University
Electrical & Computer Engineering Department
Logan, UT 84322-4120

8. PERFORMING ORGANIZATION REPORT NUMBER

Final

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFOSR/NM
110 Duncan Avenue, Room B115
Bolling AFB, DC 20332-8050

10. SPONSOR/MONITOR'S ACRONYM(S)

AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release,
distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

In separate parts, this report addresses: (1) the use of multiscale modulation to obtain bandwidth-efficient modulation, and the performance of such modulation in fading channels; (2) the analysis of transmission through fading channels, and approaches to the design of transmitter and receiver waveforms which will perform optimally in fading channels. In the first case, it is shown that using wavelet and/or scaling waveforms can lead to modulation which is comparable to, or better than, many conventional schemes, including GMSK. Performance in fading channels is similar to that of other methods. In the second part, the performance of multiscale signaling is addressed using the matched-filter bound method. The signal design problem is formulated as an optimization problem and an analysis approach outlined which leads to maximization of SNR in fading channels.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF: U			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Todd K. Moon
U	U	U	UU		19b. TELEPHONE NUMBER (Include area code) (435) 755-9230

the improvement is at best moderate. Portions of these materials (greatly abbreviated) were presented at the International Telemetering Conference (ITC) [2]. In section 6, the concept of adapting the transmitted spectrum by tuning wavelet parameters is briefly introduced.

In Part II, we consider several aspects of the analysis and design of signal waveforms for fading channels. Section 7 provides a discussion of modeling for transmission over fading channels, describes some fading channel models, and provides a literature survey or related work. This work has not yet appeared in print. In section 8, an analysis of transmission through a multiplicative fading channel is performed using two methods of analysis. This section is strictly a matter of analysis, undertaken to understand more clearly the nature of fading. This section has not been published elsewhere. In section 9, a method is introduced to design signal waveforms which are optimal with respect to fading without the assumption of synchronization. This design is extended in section 10 to the question of designing the signaling waveform and the receiver waveform to achieve maximum signal to noise ratio when perfect synchronization is assumed. The design methodology is outlined but, due to the fact that research is still underway, waveforms and performance results are not presented.

A set of appendixes describes some details of computations need in sections 3.1 and 4.

In summary, this research summarized in this report offers the following contributions:

1. A model for using wavelet waveforms and employing their shift and scale orthogonality properties to obtain flexible, bandwidth-efficient modulation waveforms.

2. An analysis of the performance of these signals in Rician fast fading channels with differential detection, showing that the waveforms perform well, but with improvements that are only moderate.

3. A matched-filter bound analysis of multiscale transmission through some common channel models, showing performance comparable (and usually somewhat better) than a similar analysis for raised cosine signal waveforms.

4. An introduction to the concept that the waveforms can be modified parametrically to match the waveform to the channel spectrum.

5. An analysis of a multi-chip fading channel model.

6. A design approach for finding the optimal transmission waveform and receiver filter matched to a channel with a prescribed fading autocorrelation function.

Part I
Wavelet Modulation

2 Notation

A notation is established here which is used throughout this first part. A multiresolution analysis [3, 1, 4, 5] is formed by a sequence of closed sub-spaces V_j of $L^2(\mathbb{R})$ which are nested according to

$$\cdots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \cdots .$$

The spaces have the following pertinent properties (among others): the scale invariance property,

$$f(t) \in V_m \Leftrightarrow f(2^m t) \in V_0$$

the shift invariance property,

$$f(t) \in V_0 \Rightarrow f(t-n) \in V_0 \text{ for all } n \in \mathbb{Z},$$

and the existence of a basis function $\phi(t)$ such that

$$V_0 = \text{span}\{\phi(t-n); n \in \mathbb{Z}\}$$

and where span$\{\cdot\}$ is the linear span of the set of functions in the argument. The basis functions additionally satisfy the *shift orthogonality* property

$$\langle \phi(t-j), \phi(t-k) \rangle = \delta_{j,k}, \tag{1}$$

where the inner product is $\langle f, g \rangle = \int f(t)g(t)\,dt$ and $\delta_{j,k}$ is the usual Kronecker delta with integer arguments. The function $\phi(t)$ is known as a *scaling function* and can be shown to be a lowpass function. The shift orthogonality property of the scaling function (1) can be shown to be equivalent to the Fourier domain constraint

$$\sum_{k=-\infty}^{\infty} |\hat{\phi}(\omega + 2k\pi)|^2 = 1, \tag{2}$$

where $\hat{\phi}(\omega)$ is the Fourier transform of $\phi(t)$ [1]. The constraint (2) is equivalent to the Nyquist zero ISI criterion (see, e.g. [6, p. 561]), so the class of scaling functions includes Nyquist signals, including the square-root raised-cosine waveform.

Let W_j be the orthogonal complement of V_j in V_{j-1}; i.e.,

$$V_{j-1} = V_j \oplus W_j, \tag{3}$$

where $\oplus$ indicates the direct sum. It follows that W_j is orthogonal to $W_{j'}$ for $j \neq j'$ and that for $j < J$,

$$V_j = V_J \oplus \bigoplus_{k=0}^{J-j-1} W_{J-k} \tag{4}$$

where all the subspaces on the right are orthogonal to each other.

It can be shown that there is a function $\psi(t)$ such that

$$W_0 = \text{span}\{\psi(t-n); n \in \mathbb{Z}\}.$$

The function $\psi(t)$ is a *wavelet function* and is a bandpass signal. From the properties of the multiresolution analysis, it follows that $\phi(t) \in V_0 \subset V_{-1}$ must be a linear combination of shifts of $\phi(2t) \in V_{-1}$,

$$\phi(t) = \sum_{k=0}^{N-1} c_k \phi(2t-k) \tag{5}$$

and $\psi(t) \in W_0 \subset V_{-1}$ can be expressed as the linear combination

$$\psi(t) = \sum_{k=0}^{N-1} d_k \phi(2t-k) \tag{6}$$

for appropriately selected sequences $\{c_k\}$ and $\{d_k\}$. Let $\phi_{j,k}(t) = 2^{j/2}\phi(2^j t - k)$ and $\psi_{j,k}(t) = 2^{j/2}\psi(2^j t - k)$. Then by the properties of the multiresolution analysis,

$$\langle \phi_{j,k}(t), \psi_{l,m}(t) \rangle = 0 \qquad j \geq l \text{ and } j,k,l,m \in \mathbb{Z}$$

and

$$\langle \psi_{j,k}(t), \psi_{l,m}(t) \rangle = \delta_{j,l}\delta_{k,m}.$$

In (5), if the number of coefficients N is finite then the function has compact support. While compact support is not necessary for communication applications (e.g. the square-root raised-cosine does not have compact support), compact support does simplify some aspects of implementation and analysis. It is assumed throughout this report that the scaling function employed has support on $t \in [0, q]$. Compactly-supported scaling functions give rise to compactly-supported wavelet functions. Wavelet and scaling functions parameterized by N coefficients in (5) and (6) are said to be in the D_N family, where D represents Daubechies [1] and N must be even. The support and smoothness of the ϕ and ψ functions increase with N. Observe that the wavelet function with two coefficients is the familiar Haar function.

Let

$$\phi_T(t) = \frac{1}{\sqrt{T}}\phi(t/T)$$

and

$$\psi_T(t) = \frac{1}{\sqrt{T}}\psi(t/T).$$

These scaled functions can be used to transmit one symbol of information every T seconds according to the baseband signal model

$$s_d(t) = \sqrt{2S}g(t), \tag{7}$$

where

$$g(t) = \sum_{i \in \mathbf{Z}} a_i \phi_T(t - iT), \tag{8}$$

and S is the signal energy, $a_i \in C$, and C is some real or complex signal constellation with 2^{n_b} signal points. In (8), T is said to be the symbol time, being the time interval between transmission of adjacent symbols, even though the duration of the symbol is qT seconds. Each scaling function transmits a single symbol. By the shift orthogonality property, the outputs of a matched filter provide sufficient information to detect the received signal independently of any other symbols. This is similar, in some respects, to partial response signaling, in which the signal is spread out over several symbol times. However, unlike partial response signaling, the detector does not need to traverse a trellis or keep track of previous symbols, because of the orthogonality between signals. Like a Nyquist zero ISI signal, the sampled outputs of the matched filter are orthogonal to signals at other symbol times and on other scales.

Using the multiresolution property of (3), a signal spanning the same space can be written using basis functions ϕ_{2T} and ψ_{2T}, each with independent data streams:

$$g(t) = \sum_{i \in \mathbf{Z}} a_{\psi,i}\psi_{2T}(t - 2it) + a_{\phi,i}\phi_{2T}(t - 2iT). \tag{9}$$

If the signal amplitudes $a_{\phi,i}$ and $a_{\psi,i}$ are each drawn from constellations containing 2^{n_b} signals, then the data rate for the signal of (9) is the same as for the signal of (8), even though the symbol times are twice

as long. The multiresolution property can be applied recursively N_s times to the scaling function in (9), as suggested by (4). The transmission using N_s scales can be written as

$$g(t) = \sum_{\sigma=1}^{N_s} \sum_{i \in \mathbf{Z}} a_{\psi,\sigma,i} \psi_{2^\sigma T}(t - i2^\sigma T) + \sum_{i \in \mathbf{Z}} a_{\phi,i} \phi_{2^{N_s} T}(t - i2^{N_s} T). \tag{10}$$

Observe that if the transmission on each scale is binary, then the energy per bit is the same for bits on all scales. This means that the power per bit decreases with increasing scale length.

For the sake of the computations below, it will be convenient to introduce more compact notation. Let

$$p_\sigma(t) = \begin{cases} \psi_{2^\sigma T}(t) & 1 \le \sigma \le N_s \\ \phi_{2^{N_s} T}(t) & \sigma = N_s + 1 \end{cases}$$

be used to represent the scaling function and the wavelets on the different scales, depending on the value of σ. Throughout the report, let $\sigma' = \min(\sigma, N_s)$ and $s' = \min(s, N_s)$ denote the actual scale number and let $T_{\sigma'} = 2^{\sigma'} T$. The signal amplitudes on scale σ at the ith symbol be indexed by multiples of $2^{\sigma'}$, with

$$b_{\sigma,2^{\sigma'} i} = \begin{cases} a_{\psi,\sigma,i} & 1 \le \sigma \le N_s \\ a_{\phi,i} & \sigma = N_s + 1. \end{cases}$$

It is also convenient to let $q_\sigma = 2^\sigma q$ (the length of the support of the signal on scale σ), with $q_\sigma = q_{\sigma'}$ when $\sigma = N_s + 1$. Using these notational conventions, the signal (10) can be written as

$$g(t) = \sum_{\sigma=1}^{N_s+1} \sum_{i \in \mathbf{Z}} b_{\sigma,2^{\sigma'} i} p_\sigma(t - iT_{\sigma'}) \tag{11}$$

The signal in (11) can also be written as

$$g(t) = \sum_{\sigma=1}^{N_s+1} \sum_{i \in 2^{\sigma'} \mathbf{Z}} b_{\sigma,i} p_\sigma(t - iT) \tag{12}$$

Figure 2(a) illustrates the time-overlap of multiscale signals for the particular case when $N_s = 2$ and $q = 5$, using the notation of (12). At every instant in time $(N_s + 1)q$ symbols are being transmitted. The figure also illustrates the bit indexing convention. Figure 3 shows another point of view for multiscale transmission. Rather than focusing on the overlap across the different signals, it illustrates the intervals in which new signals are launched. The overall signal pattern in figure 3 repeats every N_s symbol times. The group of signals within a repeating pattern is called a *supersymbol*, with length $N_s T$ seconds and containing 2^{N_s} individual symbols. The constituent symbols in a supersymbol may be considered as components in a multidimensional signal space. By sending information on a subset of the full set of scales, a variety of signal dimensionalities may be obtained.

It is interesting to examine the spectral efficiency of multiscale signaling. Due to the multiresolution property of the scaling functions, it can be shown [7] that if the signal constellation on each scale is the same, the power spectrum of the multiscale signal (11) is the same as the power spectrum of the single-scale signal (8). The spectral efficiency of the signal may be determined by means of the fractional out-of-band power (FOOBP), defined by

$$\eta(B) = 1 - \frac{\int_{-B/2}^{B/2} G(f) df}{\int_{-\infty}^{\infty} G(f) df}$$

where $G(f)$ is the power spectral density of the signal,

$$G(f) = \frac{|\hat{\phi}(2\pi f)|^2}{T}$$

and $\hat{\phi}(2\pi f)$ is the Fourier transform of $\phi(t)$. Figure 4 illustrates the FOOBP for transmission using scaling functions from the D_4, D_6, D_8, and D_{10} families. Also shown, for comparison, is the FOOBP for flat-topped pulse signaling, MSK, SFSK, and GMSK with $h = 0.36$ [8]. For each waveform, 4-PSK signaling (2 bits/symbol) is used. Being a continuous signal, the D_4 scaling function is more spectrally localized than the flat-topped pulsed, but not as much as MSK. Scaling functions having more coefficients, hence better regularity, have better localization. The D_6 scaling function has spectral localization similar to that of MSK, and D_8 and D_{10} do better than MSK. The longer scaling functions do as well as GMSK or SFSK or other spectrally efficient techniques. Even longer scaling functions are possible, resulting in even better spectral localization. The only performance penalty for going with longer signals is a slight increase in latency at the matched filter, and the need for filters matched at every scale. In interpreting the FOOBP plot of figure 4, the T normalizing the horizontal axis is the reciprocal of the transmission rate (in bits/sec): $T = 1/R$.

3 Fast Rician Fading Channel Model

For the purposes of analysis through the fading channel, the signal of (11) is assumed to be binary, $b_{\sigma,i} \in \pm 1$. The model for the fast fading Rician channel is shown in figure 5 [9], where

$$\tilde{s}_d(t) = \mathrm{Re}\{\sqrt{2S}g(t)\exp[j2\pi f_c t]\}$$

and $g(t)$ is given by (11). The signal $\tilde{s}_r(t)$ is the fading component, given by

$$\tilde{s}_r(t) = \mathrm{Re}[\xi(t)g(t - t_D)\exp[j2\pi(f_c + f_D)(t - t_D)] \tag{13}$$

where t_D is the delay of the fading component, f_D is the Doppler shift of the signal, and $\xi(t)$ is a complex, stationary, zero-mean Gaussian random process with autocorrelation function

$$R_\xi(\tau) = \frac{1}{2}E[\xi^*(t)\xi(t+\tau)] = D\rho_\xi(\tau).$$

D is a constant and $\rho_\xi(\tau)$ is normalized so that $\rho_\xi(0) = 1$. The baseband equivalent fading component is

$$\begin{aligned} s_r(t) &= \xi(t)g(t - t_D)\exp[j2\pi(f_D t - (f_c + f_D)t_D)] & (14)\\ &= \xi(t)\sum_{\sigma=1}^{N_s+1}\sum_{i\in\mathbb{Z}} b_{\sigma,i2^{\sigma'}}p_\sigma(t - t_D - iT_{\sigma'})\exp[j2\pi(f_D t - (f_c + f_D)t_D)]. & (15)\end{aligned}$$

The channel also introduces the AWGN baseband noise process $n_0(t)$ with correlation

$$R_n(\tau) = \frac{1}{2}E[n_0^*(t)n_0(t+\tau)] = N_0\delta(\tau).$$

The baseband equivalent received signal is

$$r(t) = s_d(t) + s_r(t) + n_0(t).$$

The receiver uses filters matched to the signal on each scale and employs differential detection. A baseband equivalent receiver is shown in figure 6(a) 6(b). The receiver for scale s correlates the received signal with a signal $p_s(t - kT)$ — the matched waveform starting at the kth signal interval for $k \in 2^{s'}\mathbb{Z}$ — and integrates over $[kT, (k + q_s)T]$.

The output of the matched filter on scale s at the sample instant is

$$z_s((k+q_s)T) \triangleq z_{s,1}(k) = c_s((k+q_s)T) + r_s((k+q_s)T) + n_s((k+q_sT)$$

where c_s is the direct signal part, r_s is the fading signal part, and n_s is the noise part. The signal part can be written as

$$c_s((k+q_s)T) \triangleq c_{s,1}(k) = \int_{kT}^{(k+q_s)T} p_s(t - kT)s_d(t)dt$$

which, due to shift and scale orthogonality, may be written as

$$c_{s,1}(k) = b_{s,k}\sqrt{2S}.$$

The noise part of the output,

$$n_s((k+q_s)T) \triangleq n_{s,1} = \int_{kT}^{(k+q_s)T} p_s(t-kT)n_0(t)dt$$

is zero-mean Gaussian with variance

$$n_{s,1} \triangleq \frac{1}{2}E[n_s^*(q_s+kT)n_s(q_s+kT)] = N_0, \tag{16}$$

due to the unit-energy of the waveform $p_s(t)$. The fading component output of the matched filter is

$$r_s((k+q_s)T) \triangleq f_{s,1}(k) = \int_{kT}^{(k+q_s)T} p_s(t-kT)s_r(t)dt.$$

For differential detection, the matched filter output at the previous sample interval is also used, which is

$$z_s((k-2^{s'}+q_s)T) = c_s((k-2^{s'}+q_s)T) + r_s((k-2^{s'}+q_s)T) + n_s((k-2^{s'}+q_s)T)$$

for $k \in 2^{s'}\mathbb{Z}$. The direct signal part is

$$c_s((k-2^{s'}+q_s)T) \triangleq c_{s,2}(k) = \int_{(k-2^{s'})T}^{(k-2^{s'}+q_s)T} p_s(t-(k-2^{s'})T)s_d(t)dt = b_{s,(k-2^{s'})}\sqrt{2S}.$$

The noise part $n_{s,2}$ again has variance N_0. The covariance between $n_{s,1}$ and $n_{s,2}$ is

$$n_{12} = \frac{1}{2}E[n_{s,1}^* n_{s,2}] = N_0 \int_0^{(q_s-2^s)T} p_s(t)p_s(t+2^{s'}T)dt = n_{21}^* = 0. \tag{17}$$

The fading component from the previous symbol instant is

$$r_s((k-2^{s'}+q_s)T) \triangleq f_{s,2}(k) = \int_{(k-2^{s'})T}^{(k-2^{s'}+q_s)T} p_s(t-(k-2^{s'})T)s_r(t)dt.$$

3.1 Probability of Error

Although the symbols at other scales and shifts are orthogonal to the symbol transmitted at scale s and interval k transmitted in the direct portion of the received signal, the random multiplicative factor and delay of the fading component $s_r(t)$ makes it so that the elements of $s_r(t)$ are not necessarily orthogonal to signal at scale s or to each other.The probability of error must account for this interference for each possible bit pattern in the interfering signals and for each alignment of the matched filter relative to the other scales.

Let $\mathcal{I}_{s,k}$ be the set of bits on other scales that overlap with the signal on scale s and starting position k and let $P_{s,k}(e|\mathcal{I}_{s,k})$ denote the probability of (bit) error for the kth position symbol at scale s, given all the interfering bits. Note that it is necessary to express the probability as a function of the symbol interval k, since amount of overlap of the matched filter on scale s with other symbols depends upon the position of the matched filter relative to the symbols on the other scales. The conditioning on the bits may be eliminated by

$$P_{s,k}(e) = \frac{1}{|\mathcal{I}_{s,k}|}\sum_{\mathcal{I}_{s,k}} P_{s,k}(e|\mathcal{I}_{s,k}), \tag{18}$$

where the sum is taken over all possible interfering bits and $|\mathcal{I}|$ denotes the cardinality of the set $\mathcal{I}$. The probability of error for symbols at scale s is obtained by averaging the probability of errors over the different matched filter alignments,

$$P_s(e) = \frac{1}{2^{N_s-s'}}\sum_{i=0}^{2^{N_s-s'}-1} P_{s,2^{s'}i}(e).$$

Finally, the overall probability of error is a combination of the probabilities on each scale, weighted by the fraction of bits in the overall transmission that are sent on each scale,

$$P(e) = \sum_{s=1}^{N_s+1} \frac{1}{2^{s'}} P_s(e). \tag{19}$$

Conditioned upon the interfering bits, the fading components of the matched filter outputs are Gaussian and the vector

$$\mathbf{v}(k|\mathcal{I}_{s,k}) \triangleq \begin{bmatrix} z_{s,1}(k) - c_{s,1}(k) \\ z_{s,2}(k) - c_{s,2}(k) \end{bmatrix} = \begin{bmatrix} f_{s,1}(k) + n_{s,1} \\ f_{s,2}(k) + n_{s,2} \end{bmatrix}$$

is Gaussian with zero mean and covariance

$$K_s(k|\mathcal{I}_{s,k}) = \frac{1}{2} E[\mathbf{v}^H(k|\mathcal{I}_{s,k})\mathbf{v}(k|\mathcal{I}_{s,k})] = \begin{bmatrix} K_{s,11}(k) & K_{s,12}(k) \\ K_{s,21}(k) & K_{s,22}(k) \end{bmatrix}.$$

Assuming that $n_0(t)$ and $\xi(t)$ are statistically independent, then

$$K_s(k|\mathcal{I}_{s,k}) = \begin{bmatrix} n_{s,11} & n_{s,12} \\ n_{s,21} & n_{s,22} \end{bmatrix} + \begin{bmatrix} r_s(k;1,1|\mathcal{I}_{s,k}) & r_s(k;1,2|\mathcal{I}_{s,k}) \\ r_s(k;2,1|\mathcal{I}_{s,k}) & r_s(k;2,2|\mathcal{I}_{s,k}) \end{bmatrix},$$

where the correlation of the fading components is

$$r_s(k;n,m|\mathcal{I}_{s,k}) = \frac{1}{2} E[r_{s,n}(k) r^*_{s,m}(k)|\mathcal{I}_{s,k}].$$

In the appendix, expressions for the correlation of the fading components are developed.

Let $\theta_{s,k}$ be the difference in angle between $z_{s,1}(k)$ and $z_{s,2}(k)$. If the bits at interval k and $k - 2^{s'}$ are the same, then a correct decision is made if $-\pi/2 < \theta_{s,k} < \pi/2$. The probability of this event not occurring is [9]

$$P_{s,k}(e|\mathcal{I}_{s,k}) = F_{s,k}(-\pi/2) - F_{s,k}(\pi/2),$$

where $F_{s,k}$ is given below. If the bits at interval k and $k - 2^s$ are of opposite sign, then a correct decision is made if $\pi/2 < \theta_{s,k} < 3\pi/2$ and the probability of error is

$$P_{s,k}(e|\mathcal{I}_{s,k}) = F_{s,k}(\pi/2) - F_{s,k}(3\pi/2) = F_{s,k}(\pi/2) - F_{s,k}(-\pi/2).$$

The function $F_{s,k}$ is defined by [10]

$$F_{s,k}(\upsilon) = \int_{-\pi/2}^{\pi/2} \frac{e^{-E_{s,k}}}{4\pi} \Big[\frac{W_{s,k} \sin(\Delta\Phi_{s,k} - \psi)}{U_{s,k} - V_{s,k} \sin t - W_{s,k} \cos(\Delta\Phi_{s,k} - \psi) \cos t} + \frac{\eta_{s,k} \sin\psi - \lambda_{s,k} \cos\psi}{1 - (\eta_{s,k} \cos\psi + \lambda_{s,k} \sin\psi) \cos t} \Big] dt \tag{20}$$

where

$$E_{s,k} = \frac{U_{s,k} - V_{s,k} \sin t - W_{s,k} \cos(\Delta\Phi_{s,k} - \psi) \cos t}{1 - (\eta_{s,k} \cos\upsilon + \lambda_{s,k} \sin\psi) \cos t}.$$

$$W_{s,k} = \sqrt{\rho_{1,s,k}\rho_{2,s,k}}$$

$$\Delta\Phi_{s,k} = \arg[c_{s,2}(k)] - \arg[c_{s,1}(k)]$$

$$U_{s,k} = \frac{1}{2}(\rho_{1,s,k} + \rho_{2,s,k}) \qquad V_{s,k} = \frac{1}{2}(\rho_{2,s,k} - \rho_{1,s,k})$$

$$\eta_{s,k} = \frac{\mathrm{Re}\{K_{s,12}(k)\}}{\sqrt{K_{s,11}(k)K_{s,22}(k)}} \qquad \lambda_{s,k} = \frac{\mathrm{Im}\{K_{s,12}(k)\}}{\sqrt{K_{s,11}(k)K_{s,22}(k)}}$$

$$\rho_{1,s,k} = \frac{|c_{s,1}(k)|^2}{2K_{s,11}(k)} \qquad \rho_{2,s,k} = \frac{|c_{s,2}(k)|^2}{2K_{s,22}(k)}.$$

For the binary differential signaling described in [9], substantial simplifications result in these equations. However, for multiscale signaling, such simplification is not possible so the results presented below are obtained by numerical integration of (20).

To compute $F_{s,k}$ it is necessary to know the correlation matrix $K_s(k|\mathcal{I}_{s,k})$ which requires knowing the correlations of the fading components. Expressions for the fading components are given in (79) and (80) in the appendix. Due to the complexity of the expression for the correlation of the fading component and the number of potential interfering bits $\mathcal{I}_{s,k}$, it is not computationally feasible to compute the correlation for every possible interfering bit pattern. An approximation is therefore made to the probability of error by averaging the value of $F_{s,k}$ obtained over 500 randomly generated bit patterns. (It was determined by simulation that the probability of error calculated by averaging $F_{s,k}$ obtained from 500 and 10000 randomly generated bit patterns are basically the same.)

Plots of probability of error have been generated to demonstrate the performance of the wavelets. In these plots, the correlation function is for a land-mobile fading channel, $R_\xi(\tau) = DJ_0(2\pi B_D\tau)$. The particular case of $f_d = 0$, $t_D = 0$, and $S/D = 10$ dB is shown in the probability of error plots. As a summary, there are four general observations. First, for both small and large B_DT, single scale wavelet signalings (signalings only with the scaling functions) generally perform better than conventional (single-scale flat-topped pulse, as reported in [9]) signal. Second, there is very little change in the probability of error among single scale wavelet signals with different support. Third, the penalty for using multiscale signaling, in term of increase in probability of error, reduces for using wavelets with longer support. And fourth, the penalty from one scale to next diminishes with the increases of the number of scales.

Figure 7 illustrates the probability of error for $B_DT = 0.01$ for conventional and wavelet signalings. In figure 7(a), wavelet signalings are with one scale. It can be seen that there is approximately a 1 dB improvement of wavelets over conventional signaling. There is very little change in the probability of error among D_2, D_4, D_6 and D_8. Figure 7(b) illustrates the probability of error for four scales of multiscale signalings. From the plots, one can observe that D_4, D_6 and D_8 perform just as well as conventional signaling. However, there is a huge penalty for D_2 due to the increase in correlation among signals from different scales.

Figure 8 illustrates the probability of error for $B_DT = 0.2$. The basic observation is the same for figure 8(a) as for figure 7(a). In figure 8(b), it can be observed that the penalty is larger when $B_DT = 0.2$ then that is when $B_DT = 0.01$ for all the families of wavelet except for D_2. Actually, D_2 with four scales is performing better in channels with $B_DT = 0.2$ then in channel $B_DT = 0.01$ while all other wavelets are performing not as good.

Figure 9 illustrates the probability of error for $B_DT = 0.5$. A diminishing increment of penalty as the number of scales increases is observed. In figure 9(a), while D_2 with single scale has around 1 dB improvement over conventional signaling through out the range shown, all signals with one scale for other families of wavelets start out with a better performance when E_B/N_0 is low and become having not as good a performance then conventional signaling as E_B/N_0 getting bigger. The crossing begins at around $E_B/N_0 = 12$ dB, first for D_8, then for D_6, and then for D_4 at around $E_B/N_0 = 14$ dB. The above order also shows the increasing performance through different wavelet families. As for two scale signals, figure 9(b) illustrates that there is a penalty for all wavelet signalings. D_2 is performing as conventional signaling, while all other wavelets do not perform as well. D_8 being a wavelet having a longer support then D_6 suffers less performance penalty. When the number of scale increases to three, figure 9(c) shows that as the increment of penalty is the highest, D_2 becomes the worst performer. Now, the order of lower probability of error becomes conventional signaling, follows by D_8, D_4, D_6, then D_2. As the increment of penalty diminishes, the plots of probability of error for four scales signaling in figure 9(d) is nearly the same as that in figure 9(c).

From figure 10, it is observed that that the performance variation as the number of scales increases in smallest for the D_8 signaling.

3.2 Conclusions

As this report has demonstrated to this point, wavelet signals can be used for bandwidth efficient communication. Most of the benefits are obtained when single scale scaling functions are used for transmission. For bandwidth consideration, the longer the support of the wavelet, the more bandwidth efficient. For signals over the fast Rician fading channel, the performance does not seem to depend significantly upon the type of wavelet used, with shorter wavelets providing only slightly more improvement than longer wavelets. However, if multiscale wavelet communication is contemplated, there is no change in bandwidth efficiency over single scale wavelet communication. But for the fading channel, there is a penalty in terms of increasing probability of error for multiscale signaling. This penalty decreases as wavelets with longer support are used. At the same time, there is also diminishes with the increment of the penalty when the number of scales is higher.

Because the improvement is only modest, there is motivation to consider signal designs which are optimal for fading channels. Aspects of the analysis and design methodology in this regard are thus taken up in the next Part.

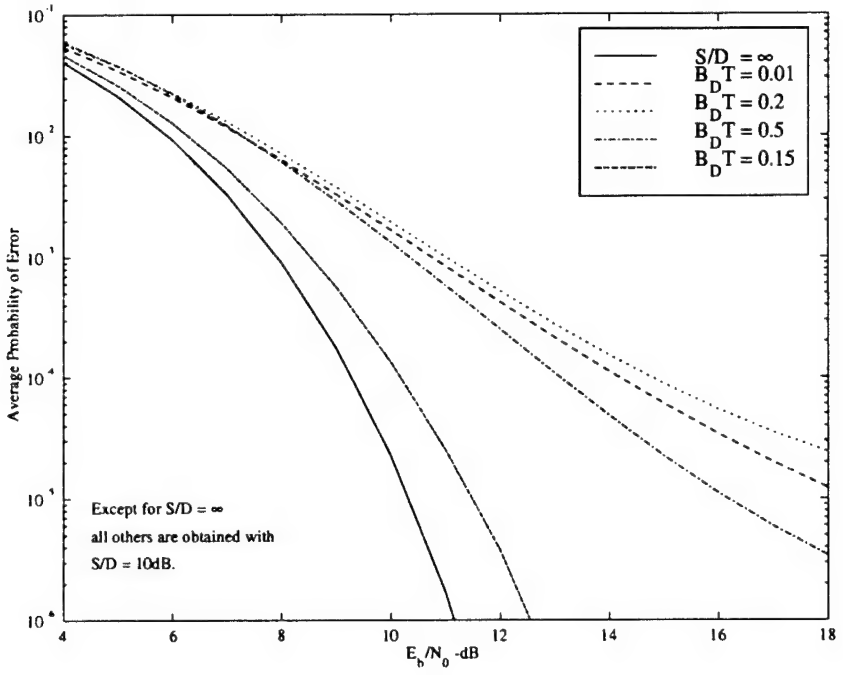


Figure 1: Probability of error for differential detection of BPSK for different fading bandwidth B_DT (from [9]).

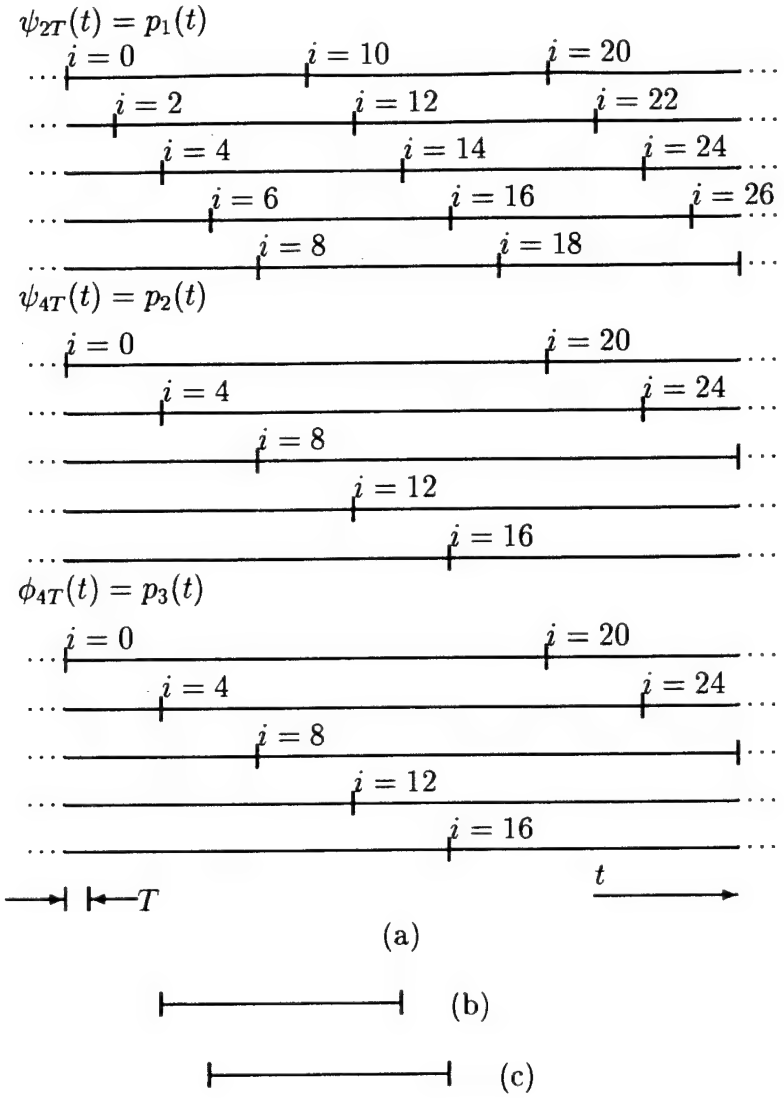


Figure 2: (a) Illustration of the time overlap of multiscale transmission for $N_s = 2$ and $q = 5$. (b) A matched filter interval for scale 1 that is scale-aligned with scale 2. (c) A non-scale-aligned matched filter interval

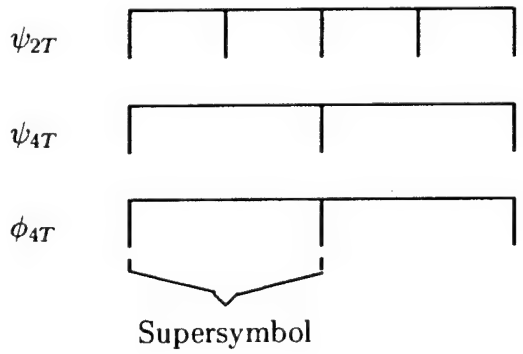


Figure 3: Relative placement in time of successive symbols on multiple scales.

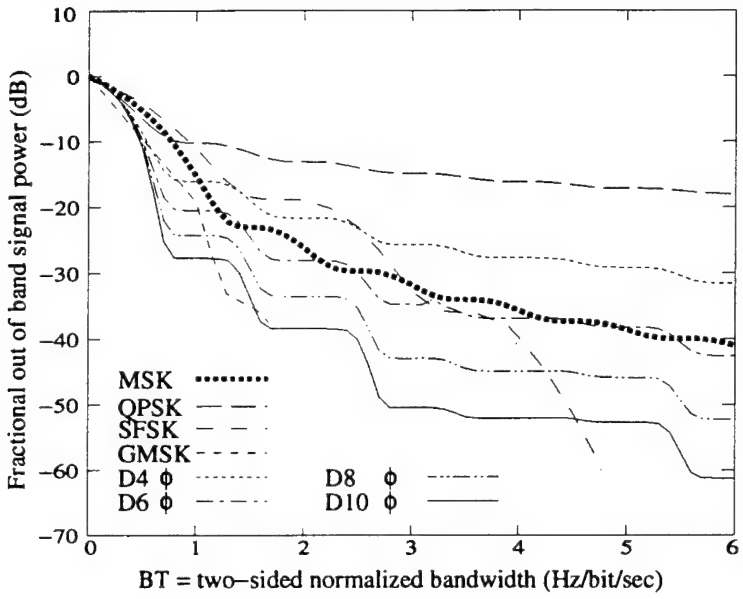


Figure 4: Fractional out-of-band power (FOOBP) for QPSK, MSK, GMSK, and scaling functions with 4, 6, 8, and 10 coefficients.

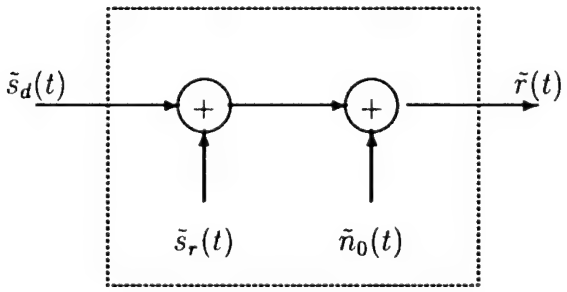
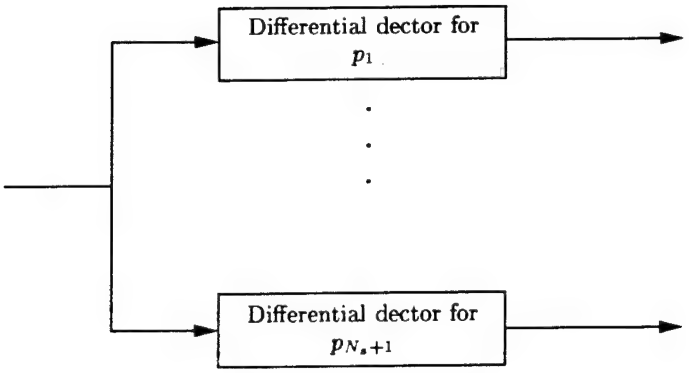
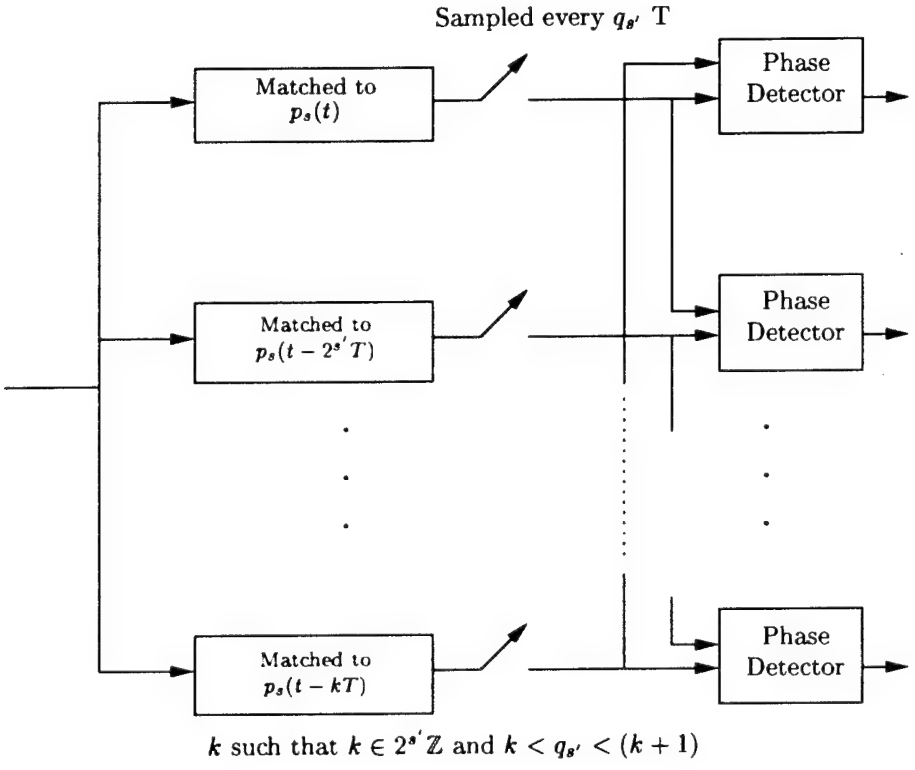


Figure 5: Rician fading channel model

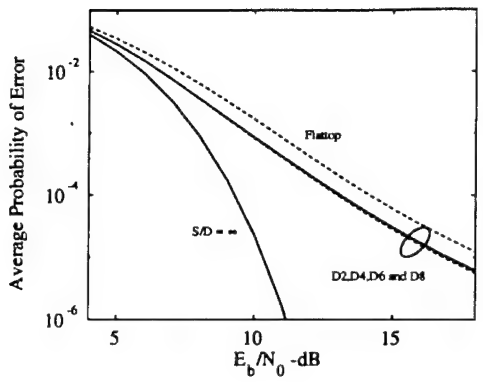


(a) Multiscale differential detector

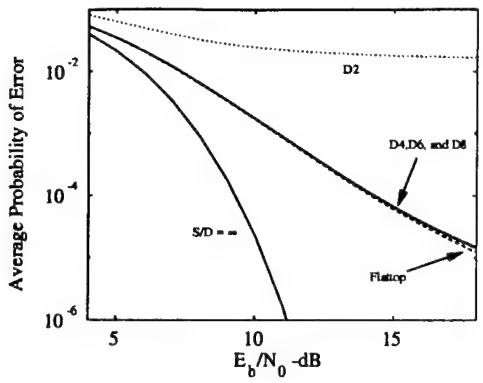


(b) Differential detector for individual p_s

Figure 6: Differential detection for multiscale signaling



(a) Comparison of probability of error for conventional signaling and single scale wavelet signalings



(b) Comparison of probability of error for conventional and wavelet signalings with four scales

Figure 7: Probability of error comparisons for $B_D T = 0.01$

4 A Matched Filter Bound for Wavelet Transmission Through a Fading Channel

In this section, an analysis is provided of the performance of multiscale transmission (wavelet modulation) using the formalism of the matched filter bound. The matched filter bound [11, 12] is a lower bound on performance, typically employed in a fading channel, which characterizes the detection capability of a single communication pulse transmitted in isolation, so that potential effects of ISI are neglected. Even though the match filter bound (MFB) is an optimal lower performance bound over communication channel that may not be practically realizable, it is a common performance measurement for communication systems [11, 12]. There are reasons for saying MFB is an optimal performance bound. First, it is assumed that all the information about the channel is known, which is not feasible for a real fading channel. Second, transmitted pulses are separated sufficiently so that no intersymbol interference (ISI) occurs. In other words, the error rate is determined by assuming only one data pulse is transmitted.

In the present case, consideration is given to a set of pulses transmitted at different scales over a common time interval, but otherwise isolated from other pulses. The MFB for multiscale wavelet pulse is derived in section 4.2. In section 4.3, the MFB is presented for examples from Daubechies' wavelet families and compare with that of raised cosine. Section 4.4 presents conclusions and discussions.

4.1 Channel Model and Wavelet Transmission

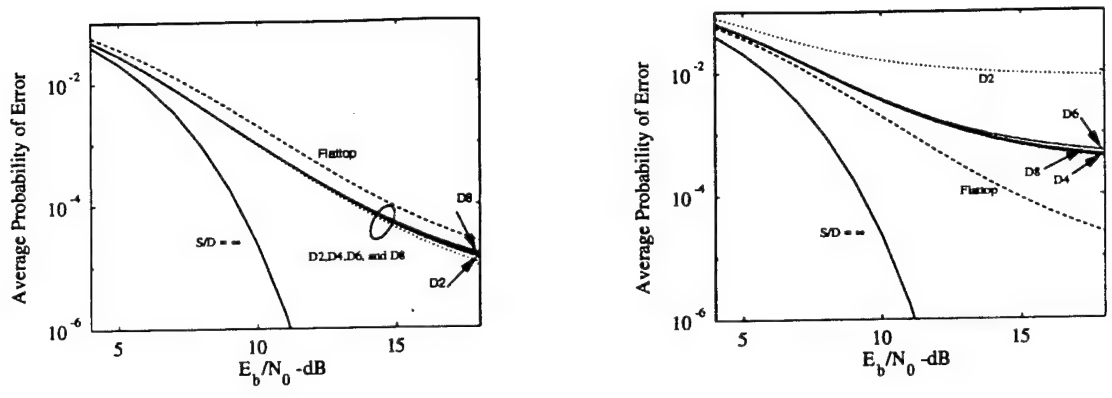
As before, the signal is modeled as

$$g(t) = \sum_{\sigma=1}^{N_s+1} b_\sigma p_\sigma(t). \tag{21}$$

For the MFB analysis of this section, only one wavelet function pulse is present on each of the different scales.

A baseband equivalent Rayleigh fading channel consisting of p paths can be represented as

$$c(t) = \sum_{i=1}^{p} \alpha_i z_i(t) \delta(t - \tau_i), \tag{22}$$



(a) Comparison of probability of error for conventional signaling and single scale wavelet signalings

(b) Comparison of probability of error for conventional and wavelet signalings with four scales

Figure 8: Probability of error comparisons for $B_D T = 0.2$

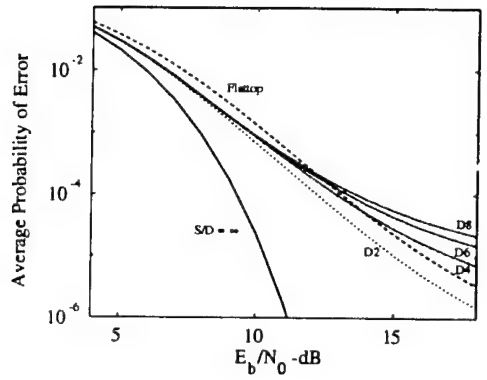
where $z_i(t)$ is a slowly time-varying, zero-mean, unit variance complex Gaussian random process, τ_i is the delay of the ith path, and α_i is the root-mean-square value of the magnitude of ith path [11]. Assuming that $z_i(t)$ does not change within the duration of a single pulse, z_i can be written in place of $z_i(t)$. The baseband signaling pulse $g(t)$ is sent through the channel (22) and corrupted by additive white Gaussian noise. At the receiver, the received signal is

$$r(t) = \sum_{i=1}^{p} \alpha_i z_i g(t - \tau_i) + n(t). \tag{23}$$

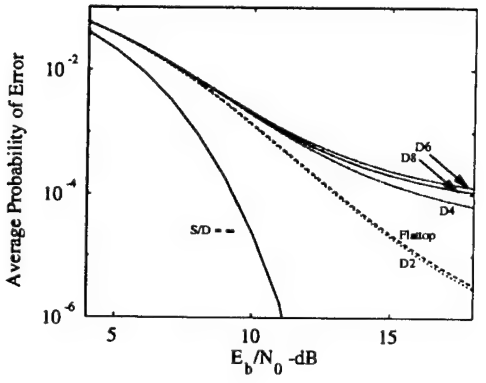
4.2 Derivation of Matched Filter Bound

To determine the data sent with multiscale signaling pulse, pass the received signal (23) through a bank of matched filters as shown in figure 11. For each scale ζ, $r(t)$ is matched to $h_{p_\zeta}(t) = \sum_{i=1}^{p} \alpha_i^* z_i^* p_\zeta^*[T_\zeta - (t - \tau_i)]$, then the signal is sampled at $t = T_\zeta$ to give us the received signal r_{p_ζ}, where T_ζ is the support of $p_\zeta(t)$.

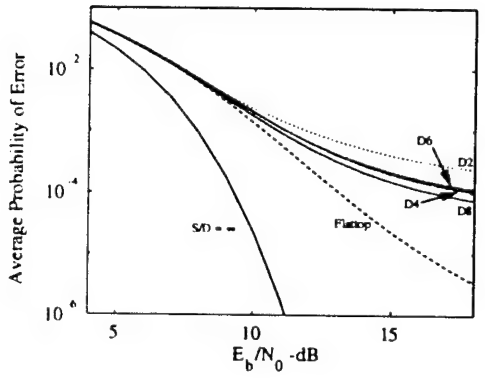
Now, let us derive the error rate $P(e_\zeta)$ for the data transmitted in scale ζ. The value of r_{p_ζ} can be



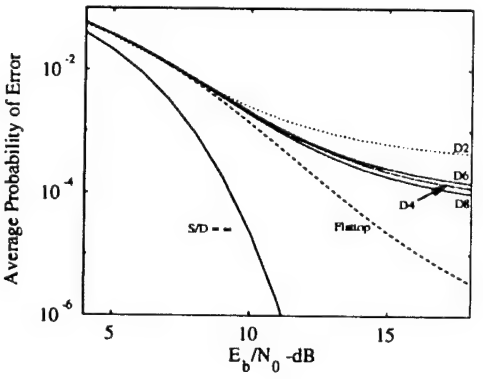
(a) Comparison of probability of error for conventional signaling and single scale wavelet signalings



(b) Comparison of probability of error for conventional and wavelet signalings with two scales



(c) Comparison of probability of error for conventional and wavelet signalings with three scales



(d) Comparison of probability of error for conventional and wavelet signalings with four scales

Figure 9: Probability of error comparisons for $B_D T = 0.5$

expressed as

$$
\begin{aligned}
r_{p_\zeta} &= \int_0^{T_\zeta} r(t) h_{p_\zeta}(T_\zeta - t)dt \\
&= \int_0^{T_\zeta} [\sum_{i=1}^{p} \alpha_i z_i g(t-\tau_i) + n(t)][\sum_{j=1}^{p} \alpha_j^* z_j^* p_\zeta^*(t-\tau_j)]dt \\
&= \int_0^{T_\zeta} [\sum_{i=1}^{p} \alpha_i z_i \sum_{\sigma=1}^{N_s+1} b_\sigma p_\sigma(t-\tau_i)][\sum_{j=1}^{p} \alpha_j^* z_j^* p_\zeta^*(t-\tau_j)]dt \\
&\quad + \int_0^{T_\zeta} n(t)[\sum_{j=1}^{p} \alpha_j^* z_j^* p_\zeta^*(t-\tau_j)]dt \\
&= \sum_{\sigma=1}^{N_s+1} \{\sum_{i=1}^{p}\sum_{j=1}^{p} \alpha_i \alpha_j^* z_i z_j^* b_\sigma \int_0^{T_\zeta} p_\sigma(t-\tau_i) p_\zeta^*(t-\tau_j)dt\} \\
&\quad + \sum_{j=1}^{p} \alpha_j^* z_j^* \int_0^{T_\zeta} n(t) p_\zeta^*(t-\tau_j)dt
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{\sigma=1}^{N_s+1} b_\sigma \sum_{i=1}^{p}\sum_{j=1}^{p} \alpha_i \alpha_j^* z_i z_j^* R_{p_\sigma p_\zeta}(\tau_i - \tau_j) + \sum_{j=1}^{p} \alpha_j^* z_j^* \int_0^{T_\zeta} n(t) p_\zeta^*(t-\tau_j)dt \\
&= \sum_{\sigma=1}^{N_s+1} b_\sigma \boldsymbol{Z}^H M_{\sigma\zeta} \boldsymbol{Z} + \sum_{j=1}^{p} \alpha_j^* z_j^* \int_0^{T_\zeta} n(t) p_\zeta^*(t-\tau_j)dt,
\end{aligned} \tag{24}
$$

where $\boldsymbol{Z} = [z_1, \cdots, z_p]^T$ and $M_{\sigma\zeta} = \{m_{\sigma\zeta_{i,j}}\}$; $m_{\sigma\zeta_{i,j}} = \alpha_i \alpha_j^* R_{p_\sigma p_\zeta}(\tau_i - \tau_j)$. Assume $b_\zeta = 1$. Let $\boldsymbol{b} = [b_1, b_2, \cdots, 1, \cdots, b_{N_s+1}]$, and the set of all possible $\boldsymbol{b}$ be $\mathcal{B}$. Note that $|\mathcal{B}| = 2^{N_s}$. Conditioning on $\boldsymbol{b}$, it is found that

$$
E(r_{p_\zeta}|\boldsymbol{b}) = \sum_{\sigma=1}^{N_s+1} b_\sigma \boldsymbol{Z}^H M_{\sigma\zeta} \boldsymbol{Z} = E_{p_\zeta}, \tag{25}
$$

and

$$
\sigma_{p_\zeta}^2 = \frac{N_0}{2} \boldsymbol{Z}^H M_\zeta \boldsymbol{Z}. \tag{26}
$$

So, the pdf of r_{p_ζ} is

$$
f(r_{p_\zeta}|\boldsymbol{b}) = \frac{1}{\sqrt{2\pi}\sigma_{p_\zeta}} \exp[-\frac{(r_{p_\zeta} - E_{p_\zeta})^2}{2\sigma_{p_\zeta}^2}]. \tag{27}
$$

For $E_{p_\zeta} \geq 0$

$$
\begin{aligned}
P(e_\zeta|\boldsymbol{b}) &= \int_{-\infty}^{0} f(r_{p_\zeta}|\boldsymbol{b}) dr_{p_\zeta} \\
&= \frac{1}{\sqrt{2\pi}\sigma_{p_\zeta}} \int_{-\infty}^{0} \exp[-\frac{(r_{p_\zeta} - E_{p_\zeta})^2}{2\sigma_{p_\zeta}^2}] dr_{p_\zeta} \\
&= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{-\frac{E_{p_\zeta}}{\sigma_{p_\zeta}}} e^{-\frac{x^2}{2}} dx \\
&= Q\left[\frac{E_{p_\zeta}}{\sigma_{p_\zeta}}\right].
\end{aligned} \tag{28}
$$

For $E_{p_\zeta} < 0$, by the same argument,

$$P(e_\zeta|\boldsymbol{b}) = 1 - Q\left[\frac{E_{p_\zeta}}{\sigma_{p_\zeta}}\right]. \tag{29}$$

Also, if $b_\zeta = -1$,

$$P(e_\zeta|\boldsymbol{b}) = \begin{cases} Q\left[\frac{E_{p_\zeta}}{\sigma_{p_\zeta}}\right] & E_{p_\zeta} \leq 0 \\ 1 - Q\left[\frac{E_{p_\zeta}}{\sigma_{p_\zeta}}\right] & \text{otherwise} \end{cases} \tag{30}$$

The average bit error probability given $\boldsymbol{b}$ can be found by averaging $P(e_\zeta|\boldsymbol{b})$ over all E_{p_ζ}:

$$P_{ave_\zeta|\boldsymbol{b}} = \int_0^\infty P(e_\zeta|\boldsymbol{b}) p(E_{p_\zeta}) dE_{p_\zeta}. \tag{31}$$

Conditioning on $\boldsymbol{b}$ is eliminated by

$$P_{ave_\zeta} = \frac{1}{|\mathcal{B}|} \sum_{\boldsymbol{b}\in\mathcal{B}} P_{ave_\zeta|\boldsymbol{b}}. \tag{32}$$

Finally, as one bit is sent through each of the scales, the average error rate over the matched filter bank is

$$P_{ave} = \frac{1}{N_s + 1} \sum_{\zeta=1}^{N_s+1} P_{ave_\zeta}. \tag{33}$$

4.3 Examples

In this section, the results of the previous section are used to derive the MFBs of some multiscale wavelet signalings for some practical fading multipath models. The channel models used are typical urban (TU) and hilly-terrain (HT), which are used as benchmarks in the Pan-European Digital Cellular (GSM) standard [11]. The wavelet functions used here are from Daubechies' wavelet families, in particular, D_4, D_6 and D_8 are used, where for D_n, n is the number of coefficients used to generate the wavelet functions [1]. These MFSs will be compared with traditional square-root raised cosine signaling with excess bandwidth $\beta = 0.35$ as specified by IS-54 [11]. The delay and amplitude profiles of the 6-path Rayleigh fading channel models are listed in Table 1. Two different symbol intervals, one for IS-54 and one for GSM, are studied.

For IS-54, the symbol interval used is $40\mu s$. Figure 12 illustrates the MFB for TU channel. From the plots, it can be seen that all MFBs are nearly the same with wavelet signalings performing a little better than raised cosine. Among different wavelet signalings, D_4 has the best MFB, while D_8 has a MFB which is the same as that of the raised cosine. In considering the effect of different number of signaling scales, it is seen that the MFB for two scales of D_4 is similar to that of with one scale, while for D_6 and D_8, two scale signaling gives marginally better MFB over one scale signaling. Figure 13 illustrates the MFB for the HT channel. Again, the MFB for single scale wavelet signalings are similar to those of raised cosine. However, for all D_4 to D_8, two scale signaling have MFBs inferior to that of single scale signaling.

For GSM, the symbol interval used is $3.8\mu s$. As the symbol interval decreases, it can be seen by comparing figures 14 and 15 to figures 12 and 13 that there is a general improvement in probability of error. Figure 14 illustrates the MFB for TU channel. It can be seen that for one scale signaling, D_4 has a $2 - - - 3$ dB gain over raised cosine. This gain decreases when longer wavelet is used. However, as the MFBs for HT channel illustrated in figure 15, all single scale wavelet pulse perform the same as raised cosine. For both TU and HT channels with GSM symbol interval, increasing the number of scales for wavelet pulse from one to two worsens the MFB.

4.4 Conclusions and Discussions Regarding the MFB

In this section, the matched filter bound (MFB) of time-discrete multipath Rayleigh fading channels derived in [13, 11] was extended to multiscale wavelet signaling communication. From the examples presented in

section 4.3, it can be seen that for all cases, single scale wavelet pulses have better MFBs compare to raised cosine.

However, one can also notice the shortcoming of applying MFB to evaluate digital communication systems in fading channels. Due to the fundamental MFB assumption that all the information about the channel is known, the MFB can be too good compare to actual performance. As shown in the section 4.3, signaling with shorter symbol interval performance better than signaling with longer symbol interval. The above finding does not agree with the common belief that a multipath channel become more frequency selective as shorter symbol intervals are used, which can lead to a higher probability of error [14].

Table 1: Delay and Amplitude Profile of GSM Channels[11].

	TU model		HT model	
Path #	Delay(μs)	Power	Delay(μs)	Power
1	0.000	1.000	0.000	1.000
2	0.813	0.669	0.813	0.251
3	1.626	0.448	1.626	0.060
4	2.439	0.300	15.447	2.258
5	3.252	0.200	16.260	0.177
6	4.056	0.134	17.073	0.122

5 Amplitude vs. Phase Modulation

The waveform designs prescribed above are linear modulation, wherein the signal amplitudes are modulated by a set of baseband functions and information is carried in the amplitude variations. In modern communications applications, such as many telemetry channels or channels in which there are nonlinear amplification elements, however, there is frequently interest in constant-envelope signals which convey the information via phase variations. Accordingly, some effort was expended to determine how to obtain the attributes of shift- and scale-orthogonality using phase-modulated waveforms. However, no analytic method has been determined yet which yields effective design procedures.

6 Spectral Adaptivity

There are entire families of wavelet functions, each with slightly different time and spectral characteristics, but all of which possess the shift- and scale-orthogonality. An application of this wealth of waveforms is that waveforms may be tuned via a parameterization so that the signal spectrum provides a better degree of match to some specified spectrum. This moves implementation closer toward the concept of digital communication as that mode of communication where the transmitted spectrum matches the channel, rather than being simply a function of the spectrum of the message signal [15]. One of the important results of information theory is the water-filling argument specifying how transmitted power should be allocated in the spectrum as a function of the noise spectrum of the channel [16, 17]. Tunable wavelets provide a mechanism for approaching this optimal allocation.

This concept has been developed up to a point in the Masters thesis of Krishna Noru at Utah State University. Since this is available through conventional channels, a citation is provided wherein details may be found [18].

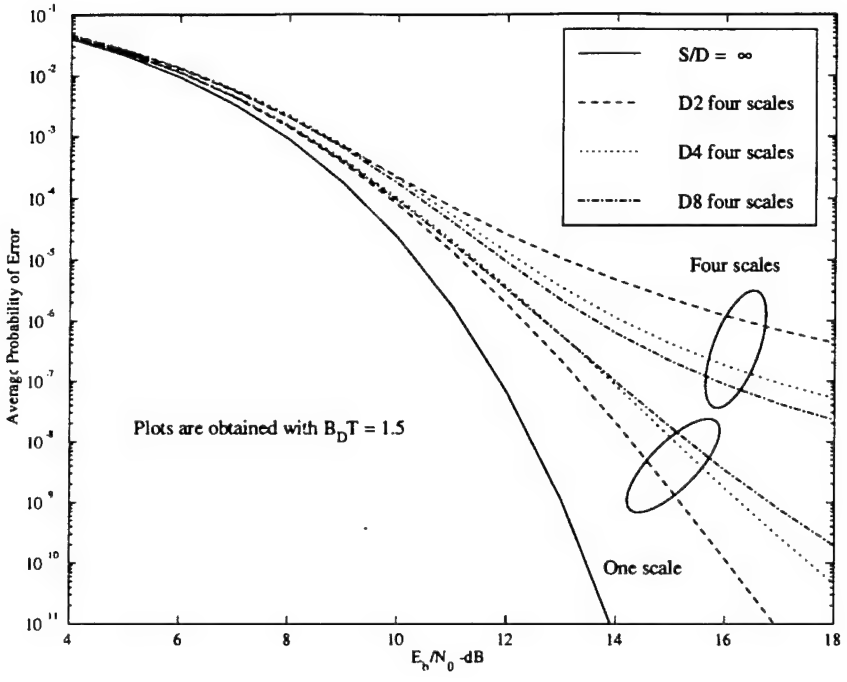


Figure 10: Illustration of probability of error penalty for multiscale wavelet signalings

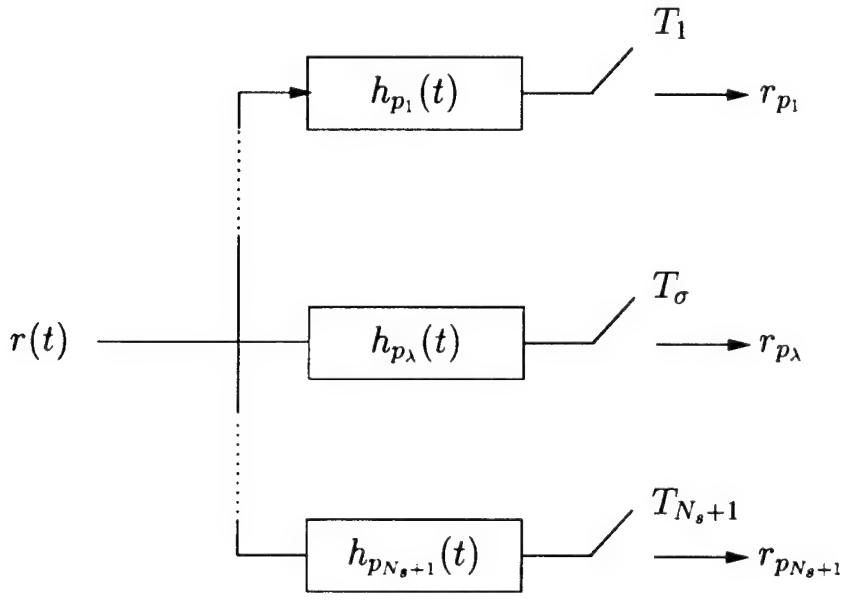


Figure 11: Matched Filter Bank for multiscale wavelet transmission

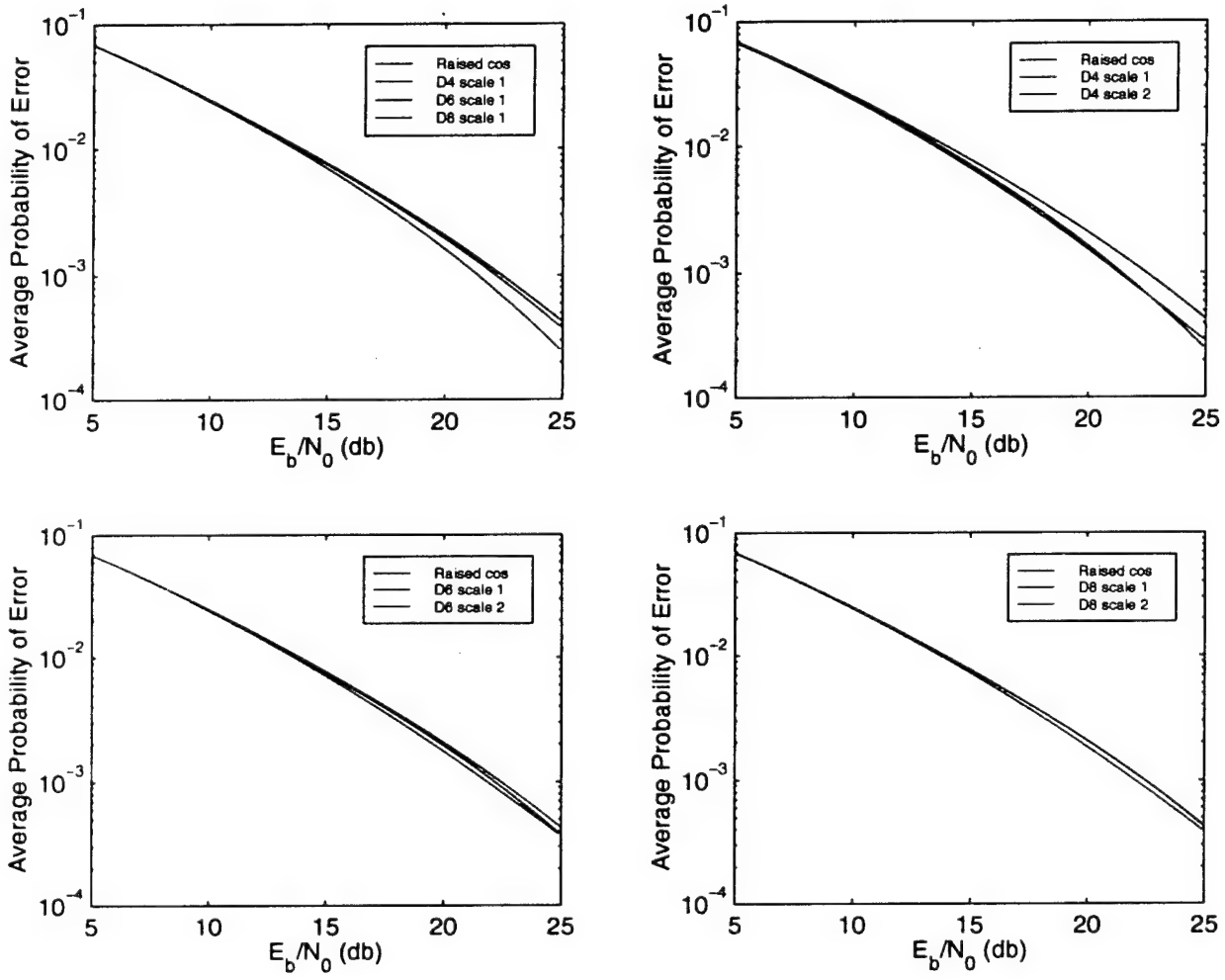


Figure 12: Typical urban with symbol interval 40μ s

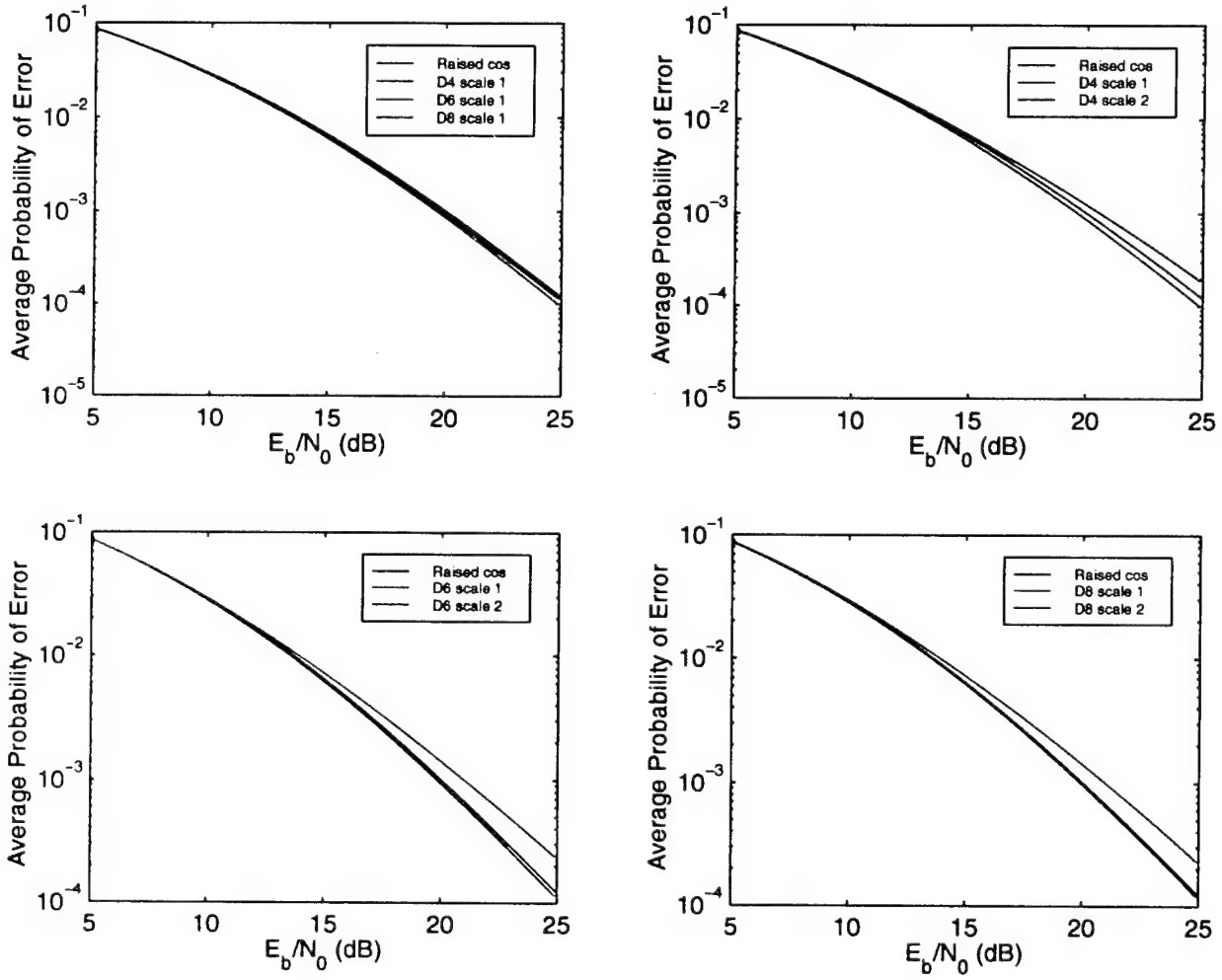


Figure 13: Hilly-terrain with symbol interval 40μ s

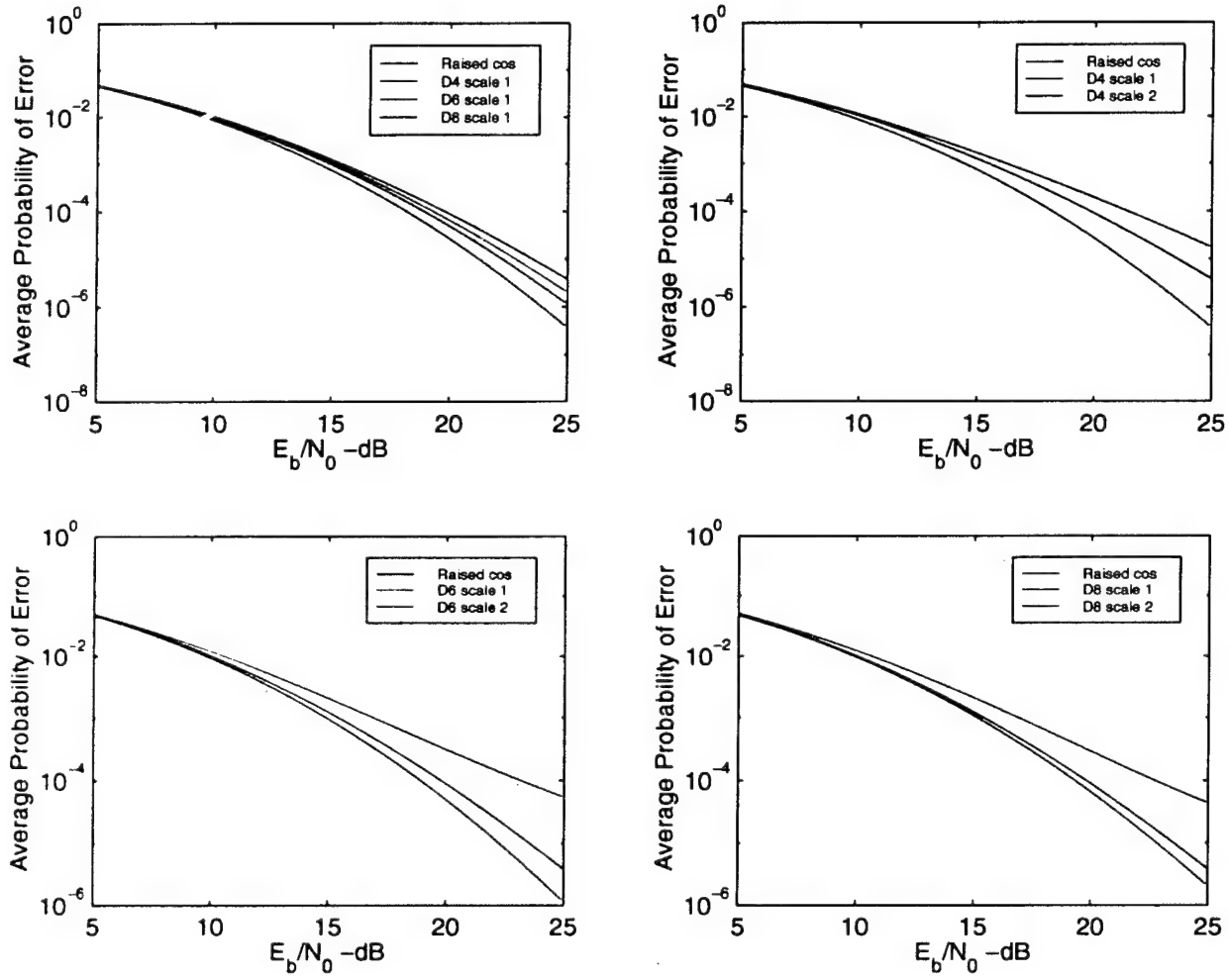


Figure 14: Typical urban with symbol interval 3.8μ s

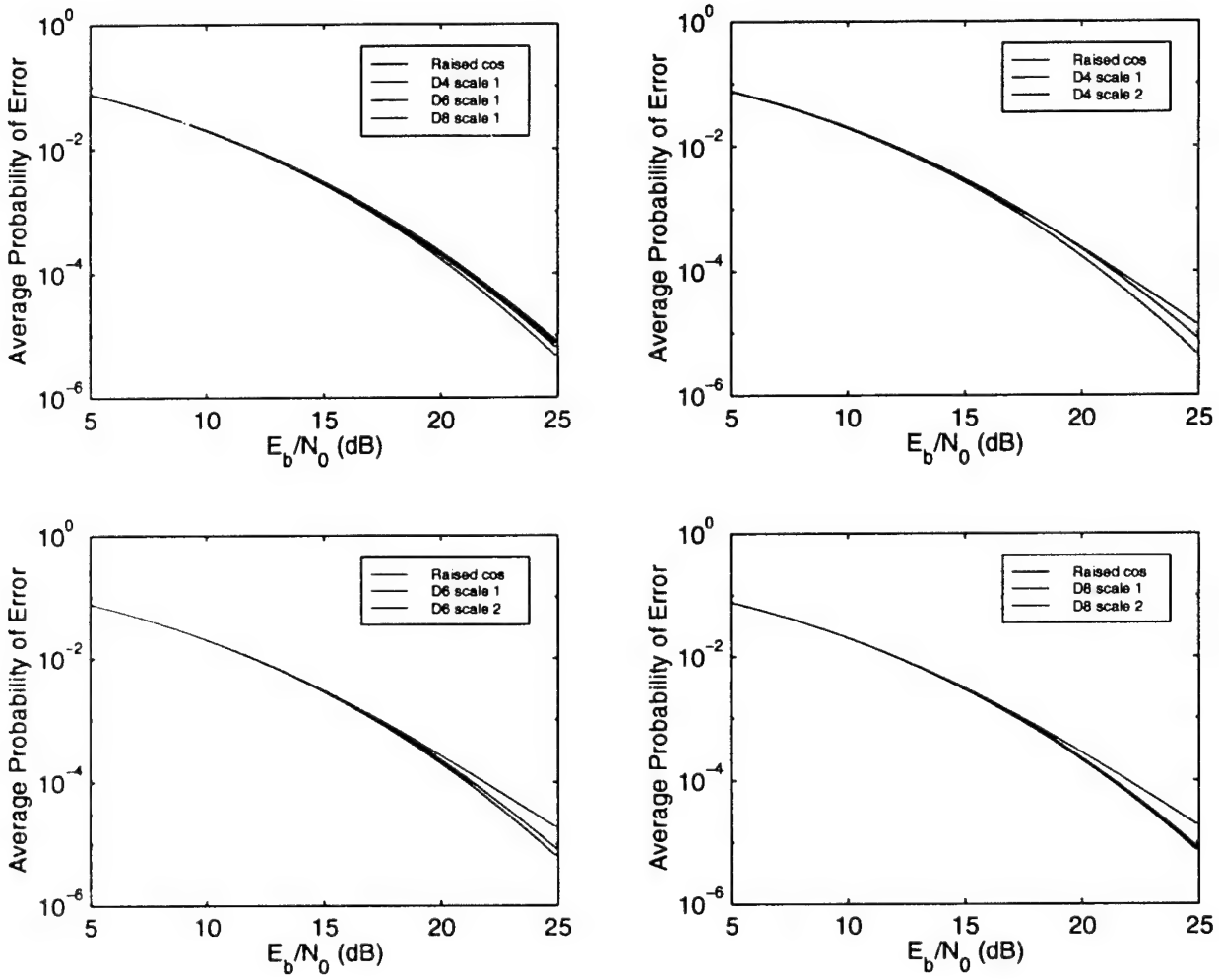


Figure 15: Hilly-terrain with symbol interval 3.8μ s

Part II
Fading Channels

7 Background and literature review

At present, wireless communication, mainly cellular, systems are only able to provide low speed data communication at error rates that are far from acceptable in wireless network connections. This is because these systems are based on technology distant from theoretical limits [19]. One of the main reasons for this draw back in technology is the hurdle of fading multipath phenomenon [20]. For example, the cellular system, GSM, we are using today, are designed more to circumvent the problems imposed by the fading multipath phenomenon than actually solving them. To avoid a too rapid fluctuation of the received signal power (caused by fading), specific signaling rate is chosen (according to the rate of fading) so that the received signal can be regarded approximately constant over a data symbol interval (slow fading assumption). At the same time, the signaling rate can not be too high as that will cause severe problems with Inter Symbol Interference, ISI (frequency selective due to multipath phenomenon) [21]. However, the good thing is there is an indication that high-speed reliable communication is possible if the user is willing to make the most of what the media would give him [22]. Because of this belief, considerable research effort has been devoted in the area of multipath fading channel communication for the past five decades [23, 24, 25, 26, 27, 28, 29, 30, 31]. This is because fading multipath phenomenon is unavoidable for high-speed wireless communication.

Usually fading multipath channels are merely called fading channels [14]. However, there are actually two basic degradations: i.e., multipaths and deep fades. The multipaths, which results from reflections, that cause transmitted signals being superimposed at the receiver. When the arrival time of the different rays are of the same order of magnitude as the duration of the transmitted signals, successive signals are smeared together, and cause ISI. The span of the excess delay is directly related to the distortion rate of the channel to the signal according to the difference in frequency. Frequency selectiveness is the term we use when the distortion rate changes a lot for a small variation in frequency. When the arrival time difference is comparable to the period of the carrier frequency, another effect results, i.e., the deep fades. The superposition of many waves of different phases give a spatial interference and cause extremely low signal power. Deep fades can also be caused by relative motion between the transmitter and the receiver, which also results in superposition interference that is called the Doppler Effect [32]. Of course, deep fades can also be caused by the continuous physical changes of the transmission media. When the interference and fluctuation are a lot more rapid compared with the signaling rate, fast fading occurs.

A lot of time, fading and multipath phenomenon are coming hand in hand. However, there are situations like stationary communications, i.e., both the transmitter and the receiver are not moving, where the occurrence of multipath transmission is the main cause of degradation of communication quality; and there are also situations like communication between high speed vehicles at an environment with no major reflection surfaces, where the main cause of degradation is fading. To make the multipath fading channel problem to be more tractable, it is helpful to consider only the multipath aspect or the fading aspect alone. In this part of the report, only the fading aspect is studied.

7.1 General Fading Channel Model

Fading channels were first modeled as large number of "scatterers" located at random points within the propagation path in 1950s and 1960s [25, 27]. This idea was primarily applied to over-the-horizon (troposcatter) communications covering a wide range of frequency bands. To reach beyond the horizon, clouds of particles in the troposphere were used as reflectors for the radio waves. The differences in path length between the large number of scattered waves give rise to Rayleigh fading if there is no dominant direct component [33].

The ground work of modeling fading channels mathematically was laid by Price [23, 24]. For the following half a century, there were tremendous amount of contributions to this field. In a recent survey paper written by Biglieri, Proakis, and Shamai, more than five hundred entries of citations reporting the state-of-the-art achievement in the research of fading channel were included in their reference [31]. By considering the absence of Prices' papers in their citation, how many more others were "missing"! Among those that were not included in their citations, there was a paper written by Bello who introduced a simple way to model

the fading phenomenon with the notion of a wide-sense stationary uncorrelated scattering (WSSUS) [26]. Uncorrelated scattering means the attenuation and phase shift of the channel associated with different path delay are uncorrelated.

A widely accepted mathematical modeling of fading channel nowaday is presented in Proakis' book [30]. A fading channel is be viewed as a continuous time-variant filter with a baseband equivalent impulse response,

$$c(\tau; t), \tag{34}$$

where $c(.;.)$ can be complex, τ is the delay in response and t is the time variable. If a signal $s(t)$ is sent through this channel with Additive Gaussian White Noise (AGWN), we will have at the receiver side,

$$r(t) = \int c(\tau; t)s(t-\tau)d\tau + n(t), \tag{35}$$

where $r(t)$ is the received signal and $n(t)$ is the AGWN.

Assuming wide-sense stationary, the autocorrelation function of $c(\tau; t)$ can be defined as

$$\phi_c(\tau_1, \tau_2; \Delta t) = \frac{1}{2}E[c^*(\tau_1; t)c(\tau_2; t+\Delta t)]. \tag{36}$$

With uncorrelated scattering,

$$\phi_c(\tau_1; \Delta t)\delta(\tau_1 - \tau_2) = \frac{1}{2}E[c^*(\tau_1; t)c(\tau_2; t+\Delta t)]. \tag{37}$$

Setting $\Delta t = 0$, the expression $\phi_C(\Delta\tau; 0)$ in (37) is usually called the multipath intensity profile or the delay power spectrum of the fading channel. Further define T_m to be the length of the span of $\phi_C(\Delta\tau; 0)$ that is essentially nonzero, and call it the multipath spread of the fading channel.

Now, if we take the Fourier transform of the channel impulse response,

$$C(f; t) = \int c(\tau; t)e^{-j2\pi f\tau}d\tau. \tag{38}$$

The corresponding autocorrelation function, with WSSUS, is

$$\begin{aligned}
\phi_C(f_1, f_2; \Delta t) &= \frac{1}{2}E[C^*(f_1; t)C(f_2; t+\Delta t)] \\
&= \frac{1}{2}\int\int E[c^*(\tau_1; t)c(\tau_2; t+\Delta t)]e^{j2\pi(f_1\tau_1 - f_2\tau_2)}d\tau_1 d\tau_2 \\
&= \int\int \phi_c(\tau_1; \Delta t)\delta(\tau_1 - \tau_2)e^{j2\pi(f_1\tau_1 - f_2\tau_2)}d\tau_1 d\tau_2 \\
&= \int \phi_c(\tau_1; \Delta t)e^{j2\pi(f_1 - f_2)\tau_1}d\tau_1 \\
&= \int \phi_c(\tau_1; \Delta t)e^{-j2\pi\Delta f\tau_1}d\tau_1 \\
&\equiv \phi_C(\Delta f; \Delta t)
\end{aligned} \tag{39}$$

Setting $\Delta t = 0$, one obtains $\phi_C(\Delta f; 0)$; it is the autocorrelation function in the frequency variable. Define the length of the span of $\phi_C(\Delta f; 0)$ that is essentially nonzero as $(\Delta f)_C$. Usually $(\Delta f)_C$ is denoted as the coherence bandwidth, which is a measure of the frequency coherence of the fading channel. To illustrate this idea, assume a signal with a bandwidth less than $(\Delta f)_C$ is transmitted through this fading channel. As all frequency components will be affected essentially similarly by the channel, this channel is said to be frequency-nonselective. However, if another signal with a bandwidth larger than $(\Delta f)_C$ is transmitted through the same fading channel, different frequency components will be affected differently by the channel. At this time, the channel is said to be frequency-selective. In this case, the signal can be severely distorted by the channel. Another useful result due to the fact that $\phi_C(\Delta\tau; 0)$ and $\phi_C(\Delta f; 0)$ are Fourier transform pairs,

$$(\Delta f)_C \approx \frac{1}{T_m}. \tag{40}$$

To study the time variation of the channel, return to $\phi_C(\Delta f; \Delta t)$ of (39). Setting $\Delta f = 0$, one essentially obtains the autocorrelation function of the channel in time averaging through all the different frequency components. Now, the length of the span of $\phi_C(0; \Delta t)$ that is essentially nonzero is defined as $(\Delta t)_C$, which is called the coherence time of the fading channel. So, for a digital communication system, if the signaling period is shorter than $(\Delta t)_C$, it may be assumed that the channel is essentially constant for individual signals and the situation of a slow fading occurs. On the other hand, if the signaling period is longer than $(\Delta t)_C$, it may no longer be assumed that the channel is constant for individual signal, and fast fading situation occurs.

In practice, $(\Delta t)_C$ is very difficult to be measured directly, so, a related function is employed. This new function is called the scattering function of the fading channel, and it is defined as

$$S(\tau; \lambda) = \int\int \phi_C(\Delta f; \Delta t) e^{-2\pi\tau\Delta f} e^{-2\pi\lambda\Delta t}\, d\Delta f\, d\Delta t, \tag{41}$$

where τ is the time delay, and λ is the Doppler frequency as the channel time variation is usually caused by physical motions of the media, the transmitter, and the receiver. The scattering function of the channel provides us with a measure of the average power output of the channel. When $\tau = 0$, the length of the span of $S(0; \lambda)$ that is essentially nonzero is called the Doppler spread B_d of the fading channel. Besides, the following relationship exists between the channel coherence time and the Doppler spread,

$$(\Delta t)_C \approx \frac{1}{B_d}. \tag{42}$$

Time variation autocorrelation function, $\phi_C(0; \Delta t)$, can be obtained from $S_C(\lambda) \equiv S(0; \lambda)$, called the Doppler power spectrum of the fading channel. Some of common Spectra and autocorrelation functions are listed in Table 2 [34, 35, 36, 37, 9].

Table 2: Spectra and autocorrelation functions of fading process

Denotation	Spectrum $S_C(\lambda)$	Autocorrelation Function $R_C(\tau)$
1. Rectangular	$\frac{1}{2B_D}$ $\quad \lvert\lambda\rvert < B_D$	$K\frac{\sin 2\pi B_D\tau}{2\pi B_D\tau}$
2. Gaussian	$K\exp(\frac{-\lambda^2}{B_D{}^2})/\sqrt{\pi}B_D$	$K\exp[-(\pi B_D\tau)^2]$
3. Land Mobile	$\frac{K}{\pi(\lambda^2 - B_D{}^2)^{1/2}}$	$KJ_0(2\pi B_D\tau)$
4. First-order Butterworth	$\frac{K}{\pi B_D(1+\lambda^2/B_D{}^2)}$	$K\exp[-2\pi B_D\lvert\tau\rvert]$

Note: $J_0(\cdot)$ is the zero-order Bessel function of the first kind and K is a constant.

7.1.1 Categories of degradation

Different fading phenomenon introduces different degradations to communication systems. Those degradations are summarized in Table 3 [14].

7.1.2 Statistical Models

Let us revisit (35), i.e.,

$$r(t) = \int c(\tau; t) s(t - \tau) d\tau + n(t).$$

Table 3: Categories of degradation according to different fading phenomenon

Frequency-nonselective	Frequency-selective	Slow fading	Fast fading
loss in SNR	ISI distortion, pulse mutilation, irreducible BER	low Doppler, loss in SNR	high Doppler, PLL failure, irreducible BER

By Parseval's relation,

$$r(t) = \int C(f;t)S(f)e^{j2\pi ft}df + n(t). \tag{43}$$

Assume there is a frequency non-selectiveness (that $C(f;t) = C(0;t)$), and

$$\begin{aligned} r(t) &= \int C(f;t)S(f)e^{j2\pi ft}df + n(t) \\ &= C(0;t)\int S(f)e^{j2\pi ft}df + n(t) \\ &= C(0;t)s(t) + n(t). \end{aligned} \tag{44}$$

The transfer function $C(0;t)$ for a frequency-nonselective fading channel may be expressed as

$$C(0;t) = \alpha(t) = a(t)e^{-j\theta(t)} \tag{45}$$

where the random process $a(t)$ represents the envelope and random process $\theta(t)$ represents the phase of the the transfer function. In this case, the channel is also called a multiplicative fading channel. Please note that a multiplicative channel can be both fast and slow fading. For different fading channels, $\theta(t)$ is always modeled as uniformly distributed over the interval $(-\pi, \pi)$. Conversely, for any fixed value of t, $a(t)$ can acquire different statistical distributions, from which different fading channel is named after. Among common distributions for $a(t)$ are the Rayleigh distribution, the Rician distribution, and the Nakagami distribution.

7.2 Literature review

All researches in engineering have two main goals: 1. understanding and 2. improvement. Research in fading channel is not an exception, two characteristics are clear in all of the works: 1. finding the ultimate performing bounds for fading channel (understanding), and 2. finding ways to combat fading effects (improvement). An interesting observation is at the early days of the study of fading channel, most of the effort was devoted to the improvement side of research, only until later, more attention was attributed to understand the matter.

The history of fading channel studying can be roughly divided into three periods. The first period spans from the early 1950's to late 1960's. The driving force behind the research about fading channel in this period was mainly the development of long-distance troposcatter communication [23, 24, 38, 33]. During this period, a tremendous effort was invested to collect real data to illustrate the fading channel, and some practical mathematical models were built [23, 26]. The single most important idea for combating fading phenomenon, i.e., "diversity", was also formed in this period [39, 40, 41, 42, 43, 44, 45, 46]. Even a lot of work was performed in the area of detection, the resulting systems were not reliable enough for some computer communication applications. This was the main reason of the introduction of coding theory to signaling on the fading channel [28].

The second period spans from early 1970's to mid 1980's. During this period, there was a drop of the interest in fading channel communications, maybe because of the high demand in the research of wire and optical communications where fading was not an important issue. Even so, there was still continuous effort in the Information and Coding Theory aspect of fading channel researches [47, 48, 49, 50].

The third period spans from the mid 1980's to present. During this period, there has been a huge increase of scientific activities in the area of fading channel research. The main driving force is the tremendous increase in the demand for personal mobile wireless communication, of which the following references typify aspects of implementations. Many new implementation schemes have been derived and reported. These approaches can be further subdivided into three sub-classes, i.e., diversity, detection and estimation, and combined methods.

1. Diversity: the application of the idea of diversity exists virtually in every branch of research of fading, either implicitly or explicitly. A basic motivation for diversity is that, under a fairly general condition, a channel affected by fading can be turned into an AWGN channel by increasing the number of diversity branches [46, 51]. There are mainly two types to diversities, i.e., time and space diversities. Time diversity is usually achieved by using some kind of rearrangement of the signaling [52], or by means of coding [53, 51, 45, 54]. Space diversity can be achieved by using multiple antenna at the transmitter and/or the receiver side [55, 56, 57].

2. Estimation and Detection: new twists to adopt old techniques to fading situations [58, 59, 60]. Application of diversity on the design of detector and estimator [61, 46]. And a lot of reports in the application of equalizers, besides, for a lot of time with robust adaptive algorithm for fast fading [62, 63, 64, 65, 66, 67, 68, 69].

3. Combined methods: Information theory and space diversity [70]. Adaptive CDMA signaling [71]. And the application of channel coding [72, 73].

8 Analysis of Multiplicative Fading Channel Model

This section considers the asymptotic behavior of a fading channel as the rate of fade increases compared to the support of signaling function when a correlation filter is employed. Because the study deals with the behavior of the channel as it is fading very fast compare to the length of the signal, it can be assumed that the support of the signal is divided into multiple chips and the signal is faded independently in each of those chips. This will greatly simplify the development of the arguments, and as the number of chips is large, it should reflect some interesting channel models.

8.1 System Model

The system model is presented in figure 16. The binary information $a = \{1, -1\}$ is carried by the signal $g(t) = as(t)$, where $\text{supp}(s(t)) = [0, T]$. Then $g(t)$ is sent through a multiplicative fading channel with a multiplying factor of $\alpha(t)$, with α being positive and Rayleigh distributed. Also, the signal is corrupted by additive Gaussian white noise $n(t)$ with variance $\sigma = \frac{N_0}{2}$. At the receiver side, the received signal is first multiplied by $s(t)$, then integrated through the support of $s(t)$ and sampled at T to give r. If $r \geq 0$ a is 1, else a is -1. The goal is to evaluate the error rate versus the transmission signal-to-noise ratio. To simplify the calculation, it is assumed that $s(t)$ is an unit energy, i.e., $\int s^2(t)dt = 1$. So, the detected signal r can be expressed as

$$r = \int_0^T \alpha(t)s^2(t)dt + n \tag{46}$$

where n is the noise component.

For fast fading situation, divide the time interval of the support of $g(t)$ into multiple subintervals, and assume the channel varies independently in each subintervals. Within any of these subintervals, it is assumed that the channel fades slowly, i.e., the multiplicative factor $\alpha_i(t)$ is constant in each subinterval, so α_i may be employed in place of $\alpha_i(t)$. The above idea is illustrated in figure 17. With the above assumption, (46) can be expressed as

$$r = \bar{r} + n$$

where

$$\bar{r} = \sum_i \alpha_i \eta_i \qquad \text{and} \qquad \eta_i = \int_{t_i}^{t_{i+1}} s^2(t)dt$$

8.2 Analysis Approach I

Consider that the interval $[0,T]$ is divided into N chips. Then the following distribution functions for the received r hold:

$$f_{\{A\}}(r|\alpha_0, \alpha_1, \cdots, \alpha_{N-1}, a = 1) = \frac{1}{\sqrt{\pi N_0}} e^{-(r-\bar{r})^2/N_0}$$

and

$$f_{\{A\}}(r|\alpha_0, \alpha_1, \cdots, \alpha_{N-1}, a = -1) = \frac{1}{\sqrt{\pi N_0}} e^{-(r+\bar{r})^2/N_0}.$$

The probability of error given $\alpha_0, \alpha_1, \cdots, \alpha_{N-1}$ and $a = 1$ is

$$\begin{aligned} p(\varepsilon|\alpha_0, \alpha_1, \cdots, \alpha_{N-1}, a = 1) &= \int_{-\infty}^{0} f_{\{A\}}(r|\alpha_0, \alpha_1, \cdots, \alpha_{N-1}, a = 1)dr \\ &= \frac{1}{\sqrt{2\pi}} \int_{\bar{r}\sqrt{\frac{2}{N_0}}}^{\infty} e^{-x^2/2} dx \\ &= \frac{1}{\sqrt{2\pi}} \int_{\frac{\bar{r}}{\sigma}}^{\infty} e^{-x^2/2} dx \end{aligned}$$

and similarly, the probability of error given $\alpha_0, \alpha_1, \cdots, \alpha_{N-1}$ and $a = -1$ is

$$\begin{aligned} p(\varepsilon|\alpha_0, \alpha_1, \cdots, \alpha_{N-1}, a = -1) &= \int_{0}^{\infty} f_{\{A\}}(r|\alpha_0, \alpha_1, \cdots, \alpha_{N-1}, a = -1)dr \\ &= \frac{1}{\sqrt{2\pi}} \int_{\frac{\bar{r}}{\sigma}}^{\infty} e^{-x^2/2} dx. \end{aligned}$$

If $a = 1$ and $a = -1$ are equally likely, then the probability of error given $\alpha_0, \alpha_1, \cdots, \alpha_{N-1}$ is

$$\begin{aligned} p(\varepsilon|\alpha_0, \alpha_1, \cdots, \alpha_{N-1}) &= \frac{1}{2} p(\varepsilon|\alpha_0, \alpha_1, \cdots, \alpha_{n-1}, a = 1) + \frac{1}{2} p(\varepsilon|\alpha_0, \alpha_1, \cdots, \alpha_{N-1}, a = -1) \\ &= \frac{1}{\sqrt{2\pi}} \int_{\frac{\bar{r}}{\sigma}}^{\infty} e^{-x^2/2} dx. \end{aligned}$$

The probability of error of this particular detection scheme may be obtained by averaging over all possible combinations of $(\alpha_0, \alpha_1, \cdots, \alpha_{N-1})$, i.e.,

$$P = \int_{\{\alpha_i\}} f_{\{A\}}(\alpha_0, \alpha_1, \cdots, \alpha_{N-1}) \left\{ \frac{1}{\sqrt{2\pi}} \int_{\frac{\bar{r}}{\sigma}}^{\infty} e^{-x^2/2} dx \right\} d(\alpha_0, \alpha_1, \cdots, \alpha_{N-1}).$$

Because $\{\alpha\}$s are independent, it follows that

$$\begin{aligned} P =& \frac{1}{\sqrt{2\pi}} \underbrace{\int_0^{\infty} \cdots \int_0^{\infty}}_{N} f_A(\alpha_0) \cdots f_A(\alpha_{N-1}) \int_{\frac{\bar{r}}{\sigma}}^{\infty} e^{-x^2/2} dx \, d\alpha_0 \, d\alpha_1 \cdots d\alpha_{N-1} \\ =& \frac{1}{\sqrt{2\pi}} \underbrace{\int_0^{\infty} \cdots \int_0^{\infty}}_{N-1} f_A(\alpha_0) \cdots f_A(\alpha_{N-2}) \\ & \left\{ \int_0^{\infty} f_A(\alpha_{N-1}) \int_{\frac{\bar{r}}{\sigma}}^{\infty} e^{-x^2/2} dx \, d\alpha_{N-1} \right\} d\alpha_0 \, d\alpha_1 \cdots d\alpha_{N-2}. \end{aligned} \tag{47}$$

Let $W_k = \sum_{i=0}^{k} \frac{\eta_i}{\sigma}\alpha_i$. Then $\frac{\bar{r}}{\sigma} = W_N = \frac{\eta_N}{\sigma}\alpha_N + W_{N-1}$, and $W_0 = 0$. Also, note that for Rayleigh distribution, the density function is $f_A(\alpha) = \frac{2\alpha}{\Omega}e^{-\alpha^2/\Omega}$ where $\Omega = E(A^2)$. Changing the order of integration as illustrated in figure 18, (47) can be rewritten as

$$
\begin{aligned}
P =& \frac{1}{\sqrt{2\pi}} \underbrace{\int_0^\infty \cdots \int_0^\infty}_{N-1} f_A(\alpha_0)\cdots f_A(\alpha_{N-2}) \left\{ \int_{W_{N-2}}^\infty e^{-x^2/2} \int_0^{\frac{\sigma(x-W_{N-2})}{\eta_{N-1}}} \frac{2\alpha_{N-1}}{\Omega} \exp(\frac{-\alpha_{N-1}^2}{\Omega}) \right. \\
& \left. d\alpha_{N-1}\, dx \right\} d\alpha_0\, d\alpha_1 \cdots d\alpha_{N-2} \\
=& \frac{1}{\sqrt{2\pi}} \underbrace{\int_0^\infty \cdots \int_0^\infty}_{N-1} f_A(\alpha_0)\cdots f_A(\alpha_{N-2}) \left\{ \int_{W_{N-2}}^\infty e^{-x^2/2} \left(1 - \exp\left[-\frac{(\frac{\sigma(x-W_{N-2})}{\eta_{N-1}})^2}{\Omega} \right] \right) dx \right\} \\
& d\alpha_0\, d\alpha_1 \cdots d\alpha_{N-2} \\
=& \frac{1}{\sqrt{2\pi}} \underbrace{\int_0^\infty \cdots \int_0^\infty}_{N-1} f_A(\alpha_0)\cdots f_A(\alpha_{N-2}) \int_{W_{N-2}}^\infty e^{-x^2/2} dx\, d\alpha_0\, d\alpha_1 \cdots d\alpha_{N-2} \\
& - \frac{1}{\sqrt{2\pi}} \underbrace{\int_0^\infty \cdots \int_0^\infty}_{N-1} f_A(\alpha_0)\cdots f_A(\alpha_{N-2}) \int_{W_{N-2}}^\infty \exp\left[-\frac{x^2}{2} - \frac{\sigma^2(x-W_{N-2})^2}{\Omega\eta_{N-1}^2} \right] dx \\
& d\alpha_0\, d\alpha_1 \cdots d\alpha_{N-2}
\end{aligned}
$$

$$
\begin{aligned}
=& \frac{1}{\sqrt{2\pi}} \int_0^\infty e^{-\frac{x^2}{2}} \left[1 - \exp\left(-\frac{\sigma^2 x^2}{\Omega\eta_0^2} \right) \right] dx - \sum_{k=0}^{N-2} A_k \\
=& \frac{1}{2} - \frac{1}{\sqrt{2\pi}} \int_0^\infty \exp\left[-\left(\frac{1}{2} + \frac{\sigma^2}{\Omega\eta_0^2} \right) x^2 \right] dx - \sum_{k=0}^{N-2} A_k \qquad (48) \\
=& \frac{1}{2} - \frac{1}{2}\sqrt{\frac{\Omega\eta_0^2}{\Omega\eta_0^2 + 2\sigma^2}} - \sum_{k=0}^{N-2} A_k,
\end{aligned}
$$

where

$$
\begin{aligned}
A_k &= \frac{1}{\sqrt{2\pi}} \underbrace{\int_0^\infty \cdots \int_0^\infty}_{k-1} f_A(\alpha_0)\cdots f_A(\alpha_k) \int_{W_k}^\infty \exp\left[-\frac{x^2}{2} - \frac{\sigma^2(x-W_k)^2}{\Omega\eta_{k+1}^2} \right] dx\, d\alpha_0\, d\alpha_1 \cdots d\alpha_k \\
&= \frac{1}{\sqrt{2\pi}} \underbrace{\int_0^\infty \cdots \int_0^\infty}_{k} f_A(\alpha_0)\cdots f_A(\alpha_{k-1}) \left\{ \int_0^\infty f_A(\alpha_k) \int_{W_k}^\infty \exp\left[-\frac{x^2}{2} - \frac{\sigma^2 x^2}{\Omega\eta_{k+1}^2} \right.\right. \\
&\quad \left.\left. + \frac{2\sigma^2 W_k x}{\Omega\eta_{k+1}^2} - \frac{\sigma^2 W_k^2}{\Omega\eta_{k+1}^2} \right] dx\, d\alpha_k \right\} d\alpha_0\, d\alpha_1 \cdots d\alpha_{k-1}.
\end{aligned}
$$

Applying the fact that

$$
W_k^2 = (\frac{\eta_k}{\sigma}\alpha_k)^2 + 2W_{k-1}\frac{\eta_k}{\sigma}\alpha_k + W_{k-1}^2,
$$

it follows that for $k \geq 1$,

$$\begin{aligned}
A_k =& \frac{1}{\sqrt{2\pi}} \underbrace{\int_0^\infty \cdots \int_0^\infty}_{k} f_A(\alpha_0) \cdots f_A(\alpha_{k-1}) \\
& \left\{ \int_0^\infty f_A(\alpha_k) \int_{W_k}^\infty \exp\left[-\frac{x^2}{2} - \frac{\sigma^2 (x - W_{k-1})^2}{\Omega \eta_{k+1}^2} + \frac{(2\sigma x - \eta_k \alpha_k - 2\sigma W_{k-1}) \eta_k \alpha_k}{\Omega \eta_{k+1}^2} \right] dx \, d\alpha_k \right\} \\
& d\alpha_0 \, d\alpha_1 \cdots d\alpha_{k-1} \\
=& \frac{1}{\sqrt{2\pi}} \underbrace{\int_0^\infty \cdots \int_0^\infty}_{k} f_A(\alpha_0) \cdots f_A(\alpha_{k-1}) \left[\int_{W_{k-1}}^\infty \exp[-\frac{x^2}{2} - \frac{\sigma^2 (x - W_{k-1})^2}{\Omega \eta_{k+1}^2}] \int_0^{\frac{\sigma(x - W_{k-1})}{\eta_k}} \right. \\
& \left. \frac{2\alpha_k}{\Omega} \exp\left(-\frac{\alpha_k^2}{\Omega} + \frac{(2\sigma x - \eta_k \alpha_k - 2\sigma W_{k-1}) \eta_k \alpha_k}{\Omega \eta_{k+1}^2} \right) d\alpha_k \, dx \right] d\alpha_0 \, d\alpha_1 \cdots d\alpha_{k-1}.
\end{aligned}$$

8.2.1 Two Chips

For the case of $N = 2$, i.e., there are two fading chips for $t \in [0, T]$, the probability of error is

$$P = \frac{1}{2} - \frac{1}{2} \sqrt{\frac{\Omega \eta_0^2}{\Omega \eta_0^2 + 2\sigma^2}} - A_0,$$

and

$$\begin{aligned}
A_0 =& \frac{1}{\sqrt{2\pi}} \int_0^\infty f_A(\alpha_0) \int_{W_0}^\infty \exp\left[-\frac{x^2}{2} - \frac{\sigma^2 x^2}{\Omega \eta_1^2} \right] dx \, d\alpha_0 \\
=& \frac{1}{\sqrt{2\pi}} \int_0^\infty \frac{2\alpha_0}{\Omega} e^{-\alpha_0^2/\Omega} \int_0^\infty \exp\left[-\frac{x^2}{2} - \frac{\sigma^2 x^2}{\Omega \eta_1^2} \right] dx \, d\alpha_0 \\
=& \frac{1}{\sqrt{2\pi}} \int_0^\infty \exp\left[-\frac{x^2}{2} - \frac{\sigma^2 x^2}{\Omega \eta_1^2} \right] \int_0^{\frac{\sigma x}{\eta_0}} \frac{2\alpha_0}{\Omega} \exp\left[-\frac{\alpha_0^2}{\Omega} + \frac{(2\sigma x - \eta_0 \alpha_k) \eta_0 \alpha_0}{\Omega \eta_1^2} \right] d\alpha_0 \, dx \\
=& \frac{1}{\sqrt{2\pi}(\eta_0^2 + \eta_1^2)} \int_0^\infty \exp\left[-\frac{x^2}{2} - \frac{\sigma^2 x^2}{\Omega \eta_1^2} \right] dx \\
& - \frac{1}{\sqrt{2\pi}(\eta_0^2 + \eta_1^2)} \int_0^\infty \exp\left[-\frac{x^2}{2} - \frac{\sigma^2 x^2}{\Omega \eta_1^2} + \frac{(\eta_0^2 - \eta_1^2) \sigma^2 x^2}{\Omega \eta_0^2 \eta_1^2} \right] dx \\
& + \frac{\eta_0 \eta_1 \sigma}{\sqrt{2\Omega}(\eta_0^2 + \eta_1^2)^{3/2}} \int_0^\infty x \exp\left[-\frac{x^2}{2} - \frac{\sigma^2 x^2}{\Omega \eta_1^2} + \frac{\eta_0^2 \sigma^2 x^2}{\Omega(\eta_1^4 + \eta_1^2 \eta_0^2)} \right] \operatorname{erf}\left(\frac{\eta_1 \sigma x}{\eta_0 \sqrt{\Omega(\eta_0^2 + \eta_1^2)}} \right) dx \\
& + \frac{\eta_0 \eta_1 \sigma}{\sqrt{2\Omega}(\eta_0^2 + \eta_1^2)^{3/2}} \int_0^\infty x \exp\left[-\frac{x^2}{2} - \frac{\sigma^2 x^2}{\Omega \eta_1^2} + \frac{\eta_0^2 \sigma^2 x^2}{\Omega(\eta_1^4 + \eta_1^2 \eta_0^2)} \right] \operatorname{erf}\left(\frac{\eta_0 \sigma x}{\eta_1 \sqrt{\Omega(\eta_0^2 + \eta_1^2)}} \right) dx,
\end{aligned}$$

where

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt.$$

From calculations on the individual parts,

$$\begin{aligned}
& \frac{1}{\sqrt{2\pi}(\eta_0^2 + \eta_1^2)} \int_0^\infty \exp\left[-\frac{x^2}{2} - \frac{\sigma^2 x^2}{\Omega \eta_1^2} \right] dx \\
=& \frac{\eta_1 \sqrt{\Omega}}{2(\eta_0^2 + \eta_1^2) \sqrt{\Omega \eta_1^2 + 2\sigma^2}},
\end{aligned}$$

and

$$\frac{1}{\sqrt{2\pi}(\eta_0^2+\eta_1^2)}\int_0^\infty \exp\left[-\frac{x^2}{2}-\frac{\sigma^2x^2}{\Omega\eta_1^2}+\frac{(\eta_0^2-\eta_1^2)\sigma^2x^2}{\Omega\eta_0^2\eta_1^2}\right]dx$$
$$=\frac{\eta_0\sqrt{\Omega}}{2(\eta_0^2+\eta_1^2)\sqrt{\Omega\eta_0^2+2\sigma^2}}.$$

Moreover, let

$$\Delta = -\frac{1}{2}-\frac{\sigma^2}{\Omega\eta_1^2}+\frac{\eta_0^2\sigma^2}{\Omega(\eta_1^4+\eta_0^2\eta_1^2)}$$
$$\Theta = \frac{\eta_1\sigma}{\eta_0\sqrt{\Omega(\eta_0^2+\eta_1^2)}}$$
$$\bar{\Theta} = \frac{\eta_0\sigma}{\eta_1\sqrt{\Omega(\eta_0^2+\eta_1^2)}},$$

and later,

$$\frac{\sqrt{2}\eta_0^2\Theta}{\sqrt{\pi}(\eta_0^2+\eta_1^2)}\int_0^\infty x\exp(\Delta x^2)\int_0^{\Theta x} e^{-t^2}dt\,dx$$
$$=\frac{\sqrt{2}\eta_0^2\Theta}{\sqrt{\pi}(\eta_0^2+\eta_1^2)}\int_0^\infty e^{-t^2}\int_{t/\Theta}^\infty x\exp(\Delta x^2)dx\,dt$$
$$=-\frac{\eta_0^2\Theta}{\sqrt{2\pi}\Delta(\eta_0^2+\eta_1^2)}\int_0^\infty \exp\left[-t^2+\frac{\Delta t^2}{\Theta^2}\right]dt \qquad \text{if } \Delta<0$$
$$=-\frac{\Theta^2\eta_0^2}{2\Delta(\eta_0^2+\eta_1^2)\sqrt{2(\Theta^2-\Delta)}},$$

and

$$\frac{\sqrt{2}\eta_1^2\bar{\Theta}}{\sqrt{\pi}(\eta_0^2+\eta_1^2)}\int_0^\infty x\exp(\Delta x^2)\int_0^{\bar{\Theta}x} e^{-t^2}dt\,dx$$
$$=\frac{\sqrt{2}\eta_1^2\bar{\Theta}}{\sqrt{\pi}(\eta_0^2+\eta_1^2)}\int_0^\infty e^{-t^2}\int_{t/\bar{\Theta}}^\infty x\exp(\Delta x^2)dx\,dt$$
$$=-\frac{\eta_1^2\bar{\Theta}}{\sqrt{2\pi}\Delta(\eta_0^2+\eta_1^2)}\int_0^\infty \exp\left[-t^2+\frac{\Delta t^2}{\bar{\Theta}^2}\right]dt \qquad \text{if } \Delta<0$$
$$=-\frac{\bar{\Theta}^2\eta_1^2}{2\Delta(\eta_0^2+\eta_1^2)\sqrt{2(\bar{\Theta}^2-\Delta)}}.$$

So, if $\Delta < 0$, the probability of error is

$$P=\frac{1}{2}-\frac{1}{2}\sqrt{\frac{\Omega\eta_0^2}{\Omega\eta_0^2+2\sigma^2}}-\frac{\eta_1\sqrt{\Omega}}{2(\eta_0^2+\eta_1^2)\sqrt{\Omega\eta_1^2+2\sigma^2}}+\frac{\eta_0\sqrt{\Omega}}{2(\eta_0^2+\eta_1^2)\sqrt{\Omega\eta_0^2+2\sigma^2}}$$
$$+\frac{\Theta^2\eta_0^2}{2\Delta(\eta_0^2+\eta_1^2)\sqrt{2(\Theta^2-\Delta)}}+\frac{\bar{\Theta}^2\eta_1^2}{2\Delta(\eta_0^2+\eta_1^2)\sqrt{2(\bar{\Theta}^2-\Delta)}}.$$

Similar techniques can be applied to cases with higher number of fading chips. However, when there are more than two chips, the calculation of $\{A_k\}$ will be tedious and difficult to trace.

8.3 Analysis Approach II

Another approach to solve the problem in finding the probability of error by using the same system model described above is to work on the characteristic functions of the distributions. For the density function

$f_A(\alpha) = \frac{2\alpha}{\Omega} e^{-\alpha^2/\Omega}$, the characteristic function is

$$\Psi_A(jv) = {}_1F_1\left(1, \frac{1}{2}; \frac{1}{4}v^2\Omega\right) + j\frac{\sqrt{\pi}}{2\sqrt{2}} v\Omega e^{-\frac{v^2\Omega}{4}},$$

where ${}_1F_1(1, \frac{1}{2}; -a)$ is the confluent hyper-geometric function satisfying

$${}_1F_1(1, \frac{1}{2}; -a) = -e^{-a}\sum_{k=0}^{\infty} \frac{a^k}{(2k-1)k!},$$

with $\bar{r} = \sum_i r_i$, where $r_i = \alpha\eta_i$, the characteristic function of the distribution of $\bar{r}$ can be found by first finding the characteristic functions for the distribution of r_i. The density function for r_i is

$$f_{R_i}(r_i) = \frac{1}{\eta_i} f_A(\frac{r_i}{\eta_i}),$$

and the characteristic function is

$$\Psi_{R_i}(jv) = {}_1F_1\left(1, \frac{1}{2}; \frac{1}{4}v^2\eta_i^2\Omega\right) + j\frac{\sqrt{\pi}}{2} v\eta_i\Omega e^{-\frac{v^2\eta_i^2\Omega}{4}}.$$

Assuming independence among the fading chips, the characteristic function of the distribution of $\bar{r}$ is equal to the product of $\{\Psi_{R_i}(jv)\}$, i.e.,

$$\Psi_{\bar{R}}(jv) = \prod_i \Psi_{R_i}(jv).$$

So, for $r = \bar{r} + n$, it follows that

$$f_R(r|\bar{r}, a = 1) = \frac{1}{\sqrt{\pi N_0}} e^{-(r-\bar{r})^2/N_0}, \qquad \text{and}$$
$$f_R(r|\bar{r}, a = -1) = \frac{1}{\sqrt{\pi N_0}} e^{-(r+\bar{r})^2/N_0},$$

which is similar to the results obtained before. In addition, the probability of error given $\bar{r}$ is

$$\begin{aligned} p(\varepsilon|\bar{r}) =& \frac{1}{\sqrt{2\pi}} \int_{\frac{\bar{r}}{\sigma}}^{\infty} e^{-x^2/2} dx \\ =& Q(\frac{\bar{r}}{\sigma}). \end{aligned}$$

Averaging over all possible $\bar{r}$,

$$\begin{aligned} P &= \int_0^{\infty} f_{\bar{R}}(\bar{r}) Q(\frac{\bar{r}}{\sigma}) d\bar{r} \\ &= \int_0^{\infty} Q(\frac{\bar{r}}{\sigma}) \cdot \frac{1}{2\pi} \int_{-\infty}^{\infty} \Psi_{\bar{R}}(jv) e^{-jv\bar{r}} dv\, d\bar{r}. \end{aligned} \tag{49}$$

Equation (49) can only be evaluated numerically. The procedure is: first, obtain the discrete samples of $\Psi_{\bar{R}}(jv)$: second, apply the DFT to get the discrete samples of $f_{\bar{R}}(\bar{r})$; third, obtain an approximation of $f_R(r)$ through the `Interpolation[]` function of `Mathematica`. Because of this, it is expected to have higher calculation error at two ends of interpolation.

8.4 Experiments and Discussion

To compare the findings with previously known results for the one-chip case, the distributions described in Proakis [30] were used. However, since Proakis did not indicate the parameters of the Rayleigh distribution, i.e., Ω, there are some difficulties in reproducing his plots. On the other hand, (48) is the same form indicated in Proakis, so it is believed that the results are the same as in Proakis. Approach I was established as the reference and plots from Approach I were used to check the reliability of the results obtained from Approach II. Another reference used was the direct evaluation of (47) for single chip case. Figure 19 shows the results from the method for a single fading chip with $\Omega = 1$ mentioned above. It can be seen that they are the same. For two chips with equal Ω, i.e., $\Omega = 1$, Approaches I and II only agree up to $S/N = 15$ dB. This is shown in figure 20, where figure 20(a) is for the result from Approach I while figure 20(b) is for the result from Approach II. Figure 20(c) shows both plots on top of each other. The deviation of the result of Approach II from Approach I, as mentioned above, is due to numerical and approximation error. It can be observed that this error is more severe for high signal-to-noise ratio. At present, it may be assumed the above fact is generalized for cases of higher number of chips.

Plots of probability of error against signal-to-noise ratio is presented in figures 19 and 20. Since the equation of probability of error were derived for single and two-chip cases, only the results from Approach II are used to formulate the observations and draw conclusions. So, all plots presented in the following are obtained from Approach II.

First, plots of density function for cases of different number of chips is shown in figure 21. In figure 21(a), $\{\eta_i\}$ is set to $\{1\}$. In figure 21(b), $\{\eta_i\}$ is set to $\{0.5, 0.5\}$. In figure 21(c), $\{\eta_i\}$ is set to $\{0.25, 0.25, 0.25, 0.25\}$. In figure 21(d), $\{\eta_i\}$ is set to $\{0.2, 0.2, 0.2, 0.2, 0.2\}$. It can be observed that the distribution become more and more Gaussian with a decreasing variance. In other words, with the model, a very Fast Fading Channel is equivalent to an additive Gaussian Channel. All these plots are obtained with $\Omega = 1$. Please note that the value of Ω affects the mean value $\bar{r}$ of the distributions. For fast fading, $\bar{r} = 1$ means simple Gaussian Channel, $\bar{r} < 1$ means a Gaussian Channel with depreciation due to fading effect. However, $\bar{r} > 1$ means an unlikely Gaussian Channel with performance gain from fast fading!

9 Aspects on Optimal Filters in Multiplicative Fading Channels Without the Assumption of Synchronization

The performance results from wavelet or multiscale signaling through a channel have not been too exciting. The results may be summarized by observing that the performance of multiscale signaling is comparable to that of other more conventional methods. From this section onward, a more generalized problem contemplated. Rather than postulating a waveform (such as a wavelet) and determining if it performs well in a fading channel, the question of finding what the optimal transmitter and receiver signals are explored. In the discussion here, some of the basic investigations are presented. Building on the approach here, other constraints can be added, such as the desirability of having the signal be localized spectrally.

For the channel model

$$r(t) = \alpha(t)g(t) + n(t),$$

where $n(t)$ is a AWGN, and where some statistic of $\alpha(t)$ is known, one would like to consider the problem of designing a filter matched to $\alpha(t)g(t)$, then shaping $g(t)$ to obtain the maximum $\frac{S}{N}$ ratio possible. That is, it is desired to find

$$\left(\frac{S}{N}\right)_{\max} = \max_{\substack{g(t) \\ \int |g(t)|^2 dt = 1}} \left\{ \max_{\substack{h(t) \\ \int |h(t)|^2 dt = 1}} \frac{\left| \int \alpha(t)g(t)h(t)dt \right|^2}{N_0} \right\} \tag{50}$$

As $\frac{S}{N}$ is maximized among all possible $g(t)$, this optimal filter problem will also be treated as a signal design problem.

The difficulty involved for solving the above optimization problem is that solutions must be obtained to integral equations, which depend on the channel statistics. Instead of working on the integral equations directly, one can use discrete approximations to reduce the complexity of (50). As the sampling interval is decreased, an asymptotic solution is approached. The discrete-time equivalent of (9) is

$$r[k] = \alpha[k]g[k] + n[k]. \tag{51}$$

This discrete filter optimization problem is actually an an eigenvalue maximization problem, which is classified as eigenfilter optimization problem. Let $f[k] = \alpha[k]g[k]$ and let R_f be the autocorrelation matrix of f, and let $\mathbf{h}$, normalized to $\mathbf{h}^H\mathbf{h} = 1$, be the desired filter. Then,

$$y[k] = \mathbf{h}^H\mathbf{f}[k].$$

The above system model is presented in figure 22.

For this system,

$$\begin{aligned}\frac{S}{N} &= \frac{\mathbf{h}^H R_f \mathbf{h}}{N_0 \mathbf{h}^H \mathbf{h}} \\ &= \frac{\mathbf{h}^H R_f \mathbf{h}}{N_0}.\end{aligned}$$

The above Rayleigh quotient is maximized by setting $\mathbf{h} = \mathbf{x}_1$ with $\mathbf{x}_1$ being the eigenvector of R_f corresponding to the largest eigenvalue λ_1 such that $R_f\mathbf{x}_1 = \lambda_1\mathbf{x}_1$. For a given R_f, the maximum $\frac{S}{N}$ possible is $\frac{\lambda_1}{N_0}$.

The optimization problem (50) can be approached by examining the random process $\alpha[k]$ with autocorrelation R_α, and finding the signaling pulse $g[k]$ with autocorrelation R_g such that the largest eigenvalue of $R_f = R_\alpha \circ R_g$ is maximized, where $A \circ B = \{a_{ij}b_{ij}\}$ is the Hadamard product of matrixes $A = \{a_{ij}\}$ and $B = \{b_{ij}\}$. Equation(50) may be re-written to reflect this observation, i.e.,

$$\left(\frac{S}{N}\right)_{\max} = \max_{\substack{g[k] \\ \sum |g[k]|^2 = 1}} \left\{\frac{\mathrm{eig}(R_\alpha \circ R_g)}{N_0}\right\}. \tag{52}$$

In the above expression, $\mathrm{eig}(A)$ returns the largest eigenvalue of A.

By observing that autocorrelation matrixes are Toeplitz matrixes, and Hadamard product of Toeplitz matrixes is also Toeplitz, Szego's Theorem can be used to help us solve the above optimization problem.

Szego's Theorem states that for R_n being an mth-order Toeplitz matrix, and $S(\omega)$ being the Fourier transform of the coefficients of R_n, and the eigenvalues of R_n are $\lambda_{n,k}$, then,

$$\lim_{n\to\infty} \frac{1}{n}\sum_{k=0}^{n-1}(\lambda_{n,k})^l = \frac{1}{2\pi}\int_0^{2\pi} S_{R_n}(\omega)^l d\omega \tag{53}$$

where l can be any positive integer [74].

For example, if $\mathbf{r} = \{r_{-m}, r_{-m+1}, \cdots, r_0, r_1, \cdots, r_m\}$, where $r_k = 0$ for $|k| > m$, then the matrix R_n is

of mth-order and

$$R_n = \begin{bmatrix} r_0 & r_{-1} & r_{-2} & \cdots & r_{-m} & & & & & \\ r_1 & r_0 & r_{-1} & r_{-2} & \cdots & r_{-m} & & & & \\ \vdots & & & & \ddots & & & & & \\ r_m & r_{m-1} & r_{m-2} & \cdots & r_0 & r_{-1} & \cdots & r_{-m} & & \\ & r_m & r_{m-1} & r_{m-2} & \cdots & r_0 & r_{-1} & \cdots & r_{-m} & \\ & & r_m & r_{m-1} & r_{m-2} & \cdots & r_0 & r_{-1} & \cdots & r_{-m} \\ \vdots & & & \ddots & & & & & & \\ & & & & & & r_m & \cdots & r_1 & r_0 & r_{-1} \\ & & & & & & & r_m & \cdots & r_1 & r_0 \end{bmatrix}. \tag{54}$$

The Fourier transform of the coefficients of R_n is

$$S_{R_n}(\omega) = \sum_{k=-m}^{m} r_k e^{-jk\omega}. \tag{55}$$

From Szego's Theorem, the distribution of the eigenvalues of R_n must be very close to that of $S_{R_n}(\omega)$ as the equality holds for any positive integer l. In other words, if entries of R_n may be altered, the values of r_k should be changed in a way that maximizes the maximum value of $S(\omega)$. Observe from (55) that

$$\begin{aligned} |S_{R_n}(\omega)| &\leq |\sum_{k=-m}^{m} r_k e^{-jk\omega}| \\ &\leq \sum_{k=-m}^{m} |r_k|. \end{aligned} \tag{56}$$

Now, consider how to find R_g for some given R_α such that the largest eigenvalue of their Hadamard product is maximized. As both R_g and R_α are autocorrelation matrices, they both are in the form of (54). It can also be assumed that the main diagonal of both matrixes are ones, and the off-diagonal entries are with values less than one. Expressing the matrices in polar form, i.e., $R_\alpha = \{|\alpha_{i,j}|e^{j\theta_{i,j}}\}$ and $R_g = \{|g_{i,j}|e^{j\phi_{i,j}}\}$, one obtains

$$S_{R_\alpha \circ R_g}(\omega) = \sum_{k=-m}^{m} (|\alpha_k|e^{j\theta_k}|g_k|e^{j\phi_k})e^{-jk\omega},$$

which is maximized when $\phi_k = -\theta_k$ and $|g_k| = 1$. At $\omega = 0$,

$$S_{R_\alpha \circ R_g}(0) = \sum_{k=-m}^{m} |\alpha_k|.$$

As all the entries of R_g are less or equal to one, the above result is the best that can be achieved.

The following observations may be made:

1. Given a fading channel for which the magnitude is varying in time as function $\alpha(t)$ with autocorrelation matrix R_α, the signaling function $g(t)$ should have an autocorrelation matrix R_g, which is in the conjugate direction of R_α.

2. $|g_k| = 1$ should be chosen to maximize the largest eigenvalue of the Hadamard product $R_\alpha \circ R_g$. This implies $g(t)$ has a autocorrelation function with constant magnitude, which is not very practical. On the other hand, (56) indicates that there is no benefit to require $g(t)$ has an autocorrelation with a support longer than that of $\alpha(t)$.

3. As some of the g_k must be such that $|g_k| < 1$, they should be allocated to those small $|\alpha_k|$s to keep $\sum_{k=-m}^{m} |\alpha_k g_k|$ as large as possible.

4. As the above guidelines only concern about the autocorrelation function of $g(t)$, which is related to its power spectrum, there is freedom to design the signal. The above understanding of signal design criteria will be helpful for orthogonal signal set design.

5. The optimal filter is the eigenfilter corresponding to the largest eigenvalue of matrix $R_\alpha \circ R_g$.

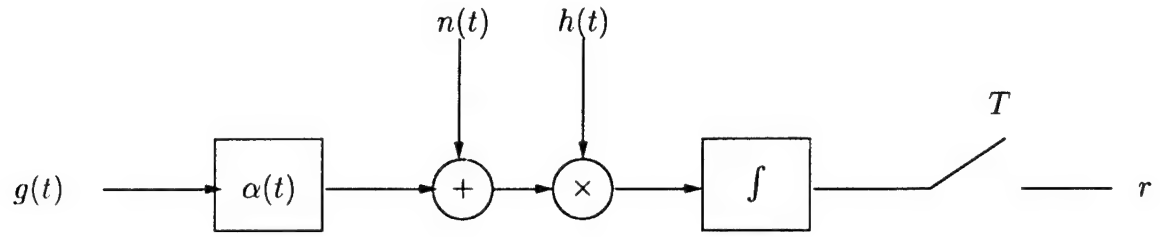


Figure 16: System model.

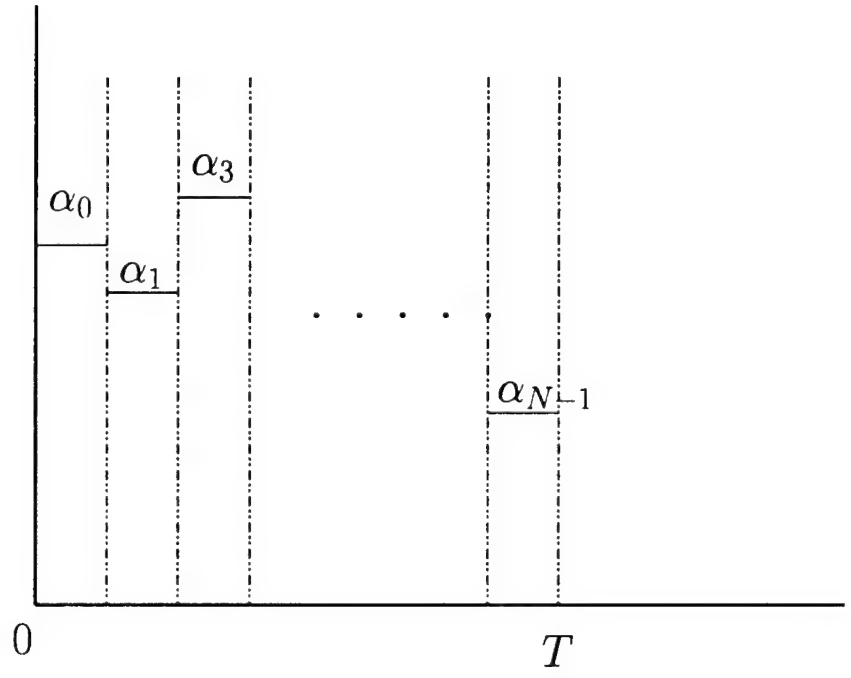


Figure 17: Independent fading chips.

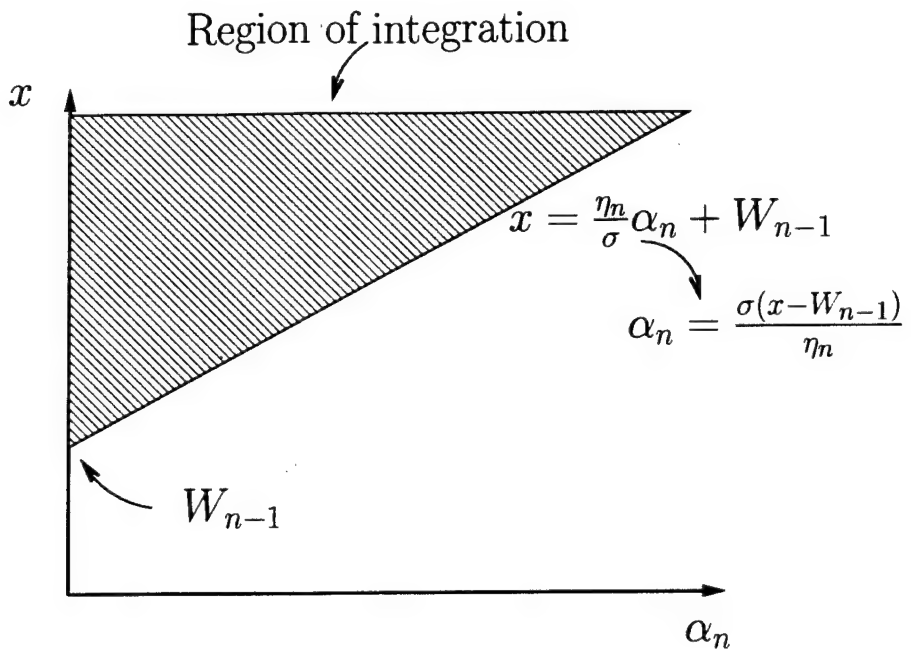


Figure 18: Change the order of integration.

10 Optimal Signal Design for Multiplicative Fading Channels With the Assumption of Perfect Synchronization

For a multiplicative channel, $\alpha(t)$ (frequency non-selective) with AWGN, the received waveform when $g(t)$ is sent can be presented as

$$r(t) = \alpha(t)g(t) + n(t), \tag{57}$$

where $n(t)$ has a two-sided PSD of $\frac{N_0}{2}$ and the autocorrelation function of $\alpha(t)$ is known. If the received waveform is passed through a correlation detector, the output of the detector consists of two separated parts at the instant of detection, i.e., the signal part and the noise part, which can be presented by the following equations.

$$r_s = \int \alpha(t)g(t)h(t)dt \qquad \text{and} \qquad r_n = \int n(t)h(t)dt. \tag{58}$$

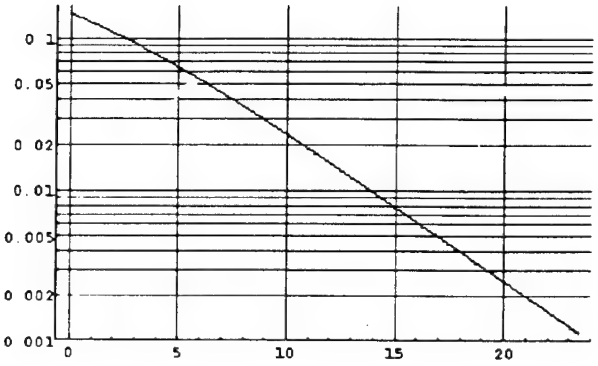
In this section, some preliminary results are presented that lead to the design of optimal signaling waveforms and optimal receiver waveforms, when the correlation function of α is known. Since these results are preliminary, the actual waveforms resulting from these designs are not presented (they are the topic of ongoing study).

One would like to study how to characterize $g(t)$ and $h(t)$ that for given $R_\alpha(\tau, \lambda) = E(\alpha(\tau)\alpha(\lambda))$, the autocorrelation function of $\alpha(t)$, the signal to noise ratio at the receiver is maximized. One further requires that $g(t)$ and $h(t)$ to be normalized, i.e.,

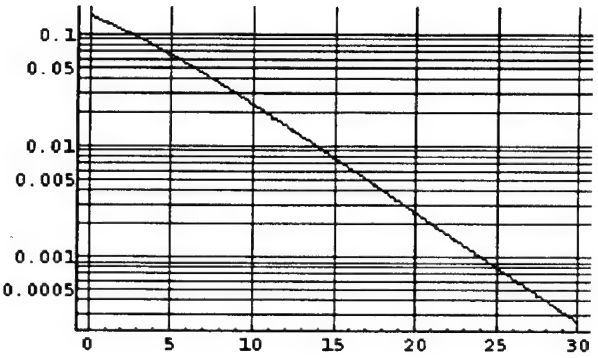
$$G(s) = \int s^2(t)dt = 1 \qquad \text{and} \qquad H(h) = \int h^2(t)dt = 1. \tag{59}$$

Maximizing the signal-to-noise ratio is equivalent to maximizing the ratio between the variance of r_s and r_n. Because of (59),

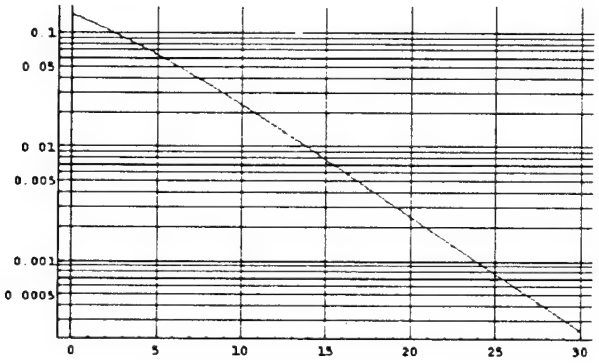
$$E(r_n^2) = \frac{N_0}{2}. \tag{60}$$



(a) Probability of error plot derived from (47)

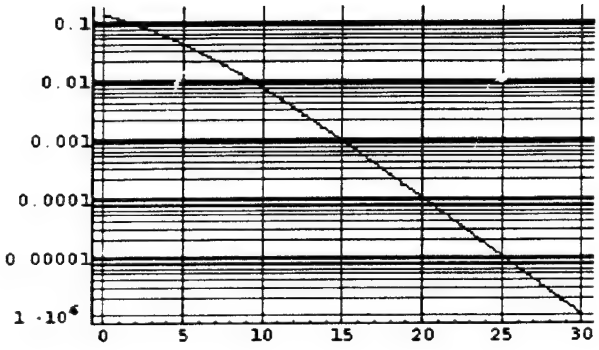


(b) Probability of error plot derived by using Approach I

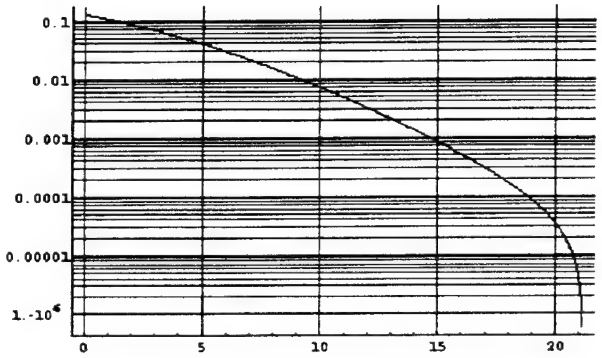


(c) Probability of error plot derived by using Approach II

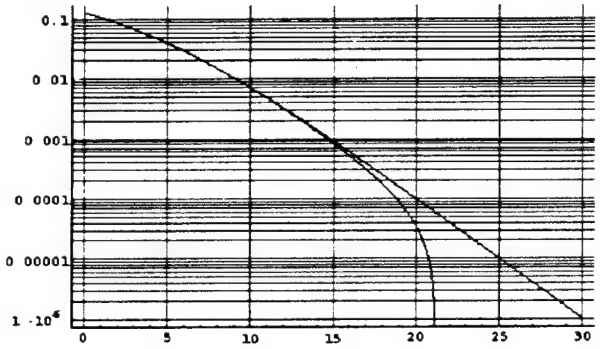
Figure 19: Comparing difference approach for one chip case



(a) Probability of error plot derived by using Approach I

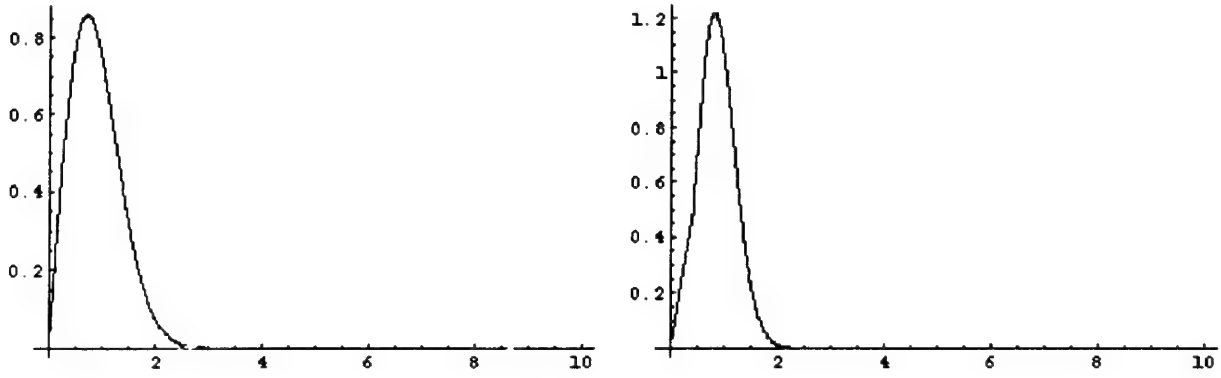


(b) Probability of error plot derived by using Approach II



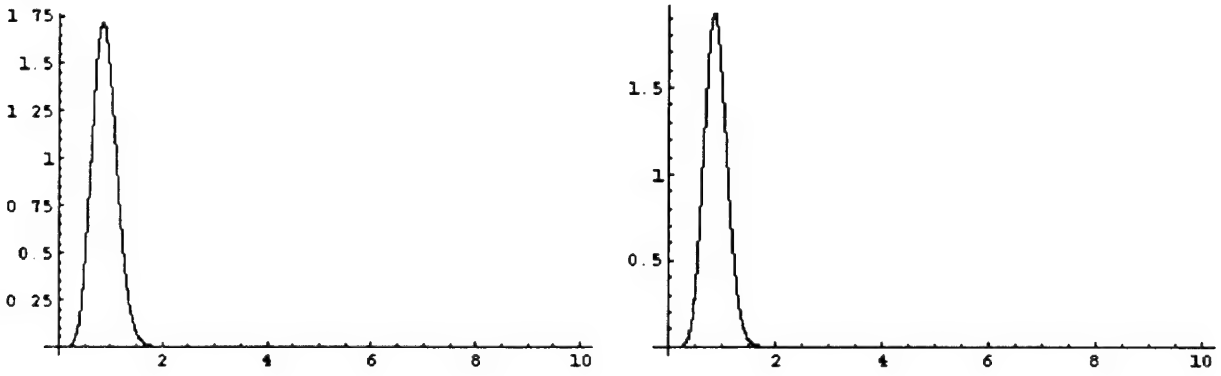
(c) Comparison of probability of error obtained from Approach I with that from Approach II

Figure 20: Comparing Approach I and II for two chips case



(a) Density function for single chip case

(b) Density function for two-chip case



(c) Density function for four-chip case

(d) Density function for five-chip case

Figure 21: Density functions for different number of chips with $\Omega = 1$

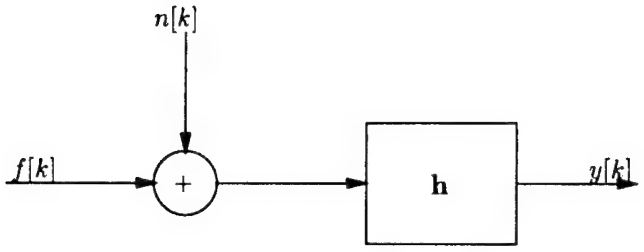


Figure 22: Noisy signal to be filtered using an eigenfilter $\mathbf{h}$ [74].

So, what one really need to do is to maximize the variance of r_s, which can be represented as the following functional

$$\begin{aligned} E(r_s^2) =& E[\int\int \alpha(\tau)g(\tau)h(\tau)\alpha(\lambda)g(\lambda)h(\lambda)d\tau d\lambda] \\ =& \int\int E[\alpha(\tau)\alpha(\lambda)]g(\tau)h(\tau)g(\lambda)h(\lambda)d\tau d\lambda \\ =& \int\int R_\alpha(\tau,\lambda)g(\tau)h(\tau)g(\lambda)h(\lambda)d\tau d\lambda \\ =& F(s,h). \end{aligned} \tag{61}$$

We use the method of Lagrange multipliers to maximize $F(s,h)$, i.e., we want to solve the equation

$$\nabla F(g_0,h_0) = \gamma_1 \nabla G(g_0) + \gamma_2 \nabla H(h_0), \tag{62}$$

where $g_0, h_0 \in \mathcal{L}^2$ (Lebesgue integrable) are the extreme points in the domain of F.

Now, for $\epsilon \in \mathbb{R}, V \in \mathcal{L}^2$,

$$\frac{\partial}{\partial s}F(g_0,h_0)\cdot V = \partial_1 F(g_0,h_0)\cdot V = \lim_{\epsilon\to 0}\frac{F(g_0+\epsilon V,h_0)-F(g_0,h_0)}{\epsilon}. \tag{63}$$

Furthermore,

$$\begin{aligned} & F(g_0+\epsilon V,h_0)-F(g_0,h_0) \\ =& \int\int R_\alpha(\tau,\lambda)(g_0+\epsilon V)(\tau)h_0(\tau)(g_0+\epsilon V)(\lambda)h_0(\lambda)d\tau d\lambda - F(g_0,h_0) \\ =& \int\int R_\alpha(\tau,\lambda)\{\epsilon V(\tau)g_0(\lambda)+\epsilon V(\lambda)g_0(\tau)\}h_0(\tau)h_0(\lambda)d\tau d\lambda \\ =& \epsilon\int\int [R_\alpha(\tau,\lambda)+R_\alpha(\lambda,\tau)]g_0(\lambda)h_0(\lambda)h_0(\tau)V(\tau)d\tau d\lambda \\ =& \epsilon\int \underbrace{\left\{\int 2R_\alpha(\tau,\lambda)g_0(\lambda)h_0(\lambda)d\lambda\right\}}_{L(\tau)} h_0(\tau)V(\tau)d\tau \\ \Rightarrow & \partial_1 F(g_0,h_0)(\tau) = L(\tau)h_0(\tau). \end{aligned} \tag{64}$$

Similarly,

$$\frac{\partial}{\partial h}F(g_0,h_0) = \partial_2 F(g_0,h_0)(\tau) = L(\tau)g_0(\tau). \tag{65}$$

So, (62) implies

$$\begin{bmatrix} L(\tau)h_0(\tau) \\ L(\tau)g_0(\tau) \end{bmatrix} = \begin{bmatrix} 2\gamma_1 g_0(\tau) \\ 2\gamma_2 h_0(\tau) \end{bmatrix} \tag{66}$$

$$\Rightarrow |L(\tau)| = 2\sqrt{|\gamma_1\gamma_2|} \quad \text{and} \quad |g_0(\tau)| = \left(\sqrt{\left|\frac{\gamma_2}{\gamma_1}\right|}\right)|h_0(\tau)|. \tag{67}$$

Because of (59), it must be the case that

$$|\gamma_1| = |\gamma_2| \quad \text{and} \quad |g_0(\tau)| = |h_0(\tau)|. \tag{68}$$

That means $g(t)$ and $h(t)$ must be similar to each other up to the same magnitude at the extrema of $F(s,h)$.

The following observations may be made:

1.

$$L(\lambda_0) = 0 \Rightarrow g_0(\lambda_0) = h_0(\lambda_0) = 0. \tag{69}$$

2.

$$\begin{aligned} F(g_0, h_0) =& \frac{1}{2} \int L(\lambda) g_0(\lambda) h_0(\lambda) d\lambda \\ =& \int \sqrt{|\gamma_1 \gamma_2|}\, g_0^2(\lambda) d\lambda \\ =& \sqrt{|\gamma_1 \gamma_2|}. \end{aligned} \tag{70}$$

3. From Tricomi [75, Chapter 3], if R_α is symmetric, then $\{\phi_k\}$, the collection of eigenfunctions of R_α, i.e.,

$$\xi_k \phi_k(\tau) = \int R_\alpha(\tau, \lambda) \phi(\lambda) d\lambda, \tag{71}$$

where ξ_k is the corresponding eigenvalue (distinctive for different k) which satisfies the orthogonality condition

$$\langle \phi_h, \phi_k \rangle = \int \phi_h(\tau) \phi_k(\tau) d\tau = 0 \quad \text{if} \quad h \neq k. \tag{72}$$

In addition, $\overline{\text{span}(\{\phi_k\})} = \mathcal{L}^2(a, b)$ with (a, b) being the support of R_α.
Now, let $f(\lambda) = g_0(\lambda) h_0(\lambda) = \sum a_k \phi_k(\lambda)$, then

$$\begin{aligned} L(\tau) =& \int 2 R_\alpha(\tau, \lambda) f(\lambda) d\lambda \\ =& \int 2 R_\alpha(\tau, \lambda) \sum a_k \phi_k(\lambda) d\lambda \\ =& \sum a_k \xi_k \phi_k(\tau). \end{aligned} \tag{73}$$

Because ϕ_k are orthogonal and ξ_k are distinct, one should be able to characterize a_k so that $|L(\tau)| =$ constant.

When the above principle of (68) is applied to a discrete equivalent of a multiplicative fading channel, one can find that optimal signals with their corresponding filters are delta functions. This makes sense because if there is a variation with the channel, one would like to allocate the transmission power to a very short instance of time so as to reduce the channel uncertainty. This also agrees with the finding of many researchers [31]. However, delta functions are not very practical for communication engineers. In this case, one need to modify the cost function in the Lagrangian (62) to require the frequency span of the signal $g(t)$ being concentrated in a small range.

11 Conclusion and Suggestion for Further Study

In this part of the report, two methods of calculating error rate for a multiplicative fading channel by assuming independent fading chips were presented. The channel statistic was also studied as the signal becomes long compared to the fading rate of the channel. It was observed that Rayleigh fading becomes more and more like a Gaussian fading with a decrease in variance. In other words, the fading effect is more and more similar to an increase in Gaussian noise.

By assuming the fading magnitude being of the same sign, it was shown that the optimal filter for any fading channels are matched filters that matches to the signaling function used. Moreover, the performance is independent to the signaling function. However, when the above assumption is lifted, the change of sign of the fading magnitude will definitely make matched filters suboptimal. As mentioned in [76], many researchers use matched filters on fast fading problems believing that matched filters are the optimal solution for this situation. The derivation in this report should be helpful to clarify the above misconception.

The correlation filter idea may be generalized to eigenfilter, which is the optimum filter for any random signal when the autocorrelation matrix of that random signal is known [74]. After employing the notion

of discrete observables suggested in [76], it is found that the filter design problem is unified with signaling function design design problem, and they are totally related to the fading channel statistic. This should give part of the explanation to Hansson's observation about the probability on the fast Rayleigh fading channel is highly dependent on the shapes of the modulator waveforms [76]. The design procedure for fading magnitude function $\alpha(t)$ with autocorrelation matrix R_α is:

1. for the case of without the synchronization assumption,
   (a) determine $g(t)$ such that, its autocorrelation matrix R_g is pointing at the conjugate direction of R_α, with entries that will keep $\sum_{k=-m}^{m} |\alpha_k g_k|$ as large as possible.
   (b) design filter $\mathbf{h}$ with value equal to the eigenvector of matrix $R_\alpha \circ R_g$ corresponding to its largest eigenvalue.
2. for the case of perfect synchronization,
   (a) require $g(t)$ to be localize in frequency domain.
   (b) construct a discrete equivalent of (62) and solve for optimal $g(t)$ and $h(t)$.

As the above optimization criteria only involve the autocorrelation function used for a given fading channel, there is room to develop good orthogonal set of signaling waveforms for fast fading channels. Of course, studies should be carried out about the affect among the orthogonal signal detectors because all of the signals will be designed to match the channel characteristic. In other words, the maximum difference in direction that the different eigenvectors point to corresponds to different signaling waveforms should be a major topic of study. This difference in angle is critical in determine the error rate.

Appendix A Computation of the Variance Components

The fading component of the matched filter output can be written as

$$r_{s,1}(k) = \sum_{\sigma=1}^{N_s+1} \sum_{i} b_{\sigma,2^{\sigma'}i} \int_{kT}^{(k+q_\sigma)T} p_s(t-kT)\xi(t)p_\sigma(t-iT_{\sigma'}-t_D)\cdot \exp[j2\pi(f_D t-(f_c+f_D)t_D)]dt, \quad k \in 2^{s'}\mathbb{Z}. \tag{74}$$

For purposes of computation, it is necessary to limit the range of i in the summation in (74) to those symbols in $s_r(t)$ which overlap with the matched filter integration interval $[kT,(k+q_s)T]$. If the matched filter for the kth symbol on scale s aligns with the start of a symbol on scale σ, the matched filter is said to be *scale-aligned*. Otherwise, the matched filter is non-scale-aligned. A matched filter interval for scale 1 which is scale-aligned with scale 2 is shown in figure 2(b) , and a non-scale-aligned matched filter interval is shown in figure 2(c). If $s \geq \sigma$, then the matched filter on scale s is always scale-aligned with scale σ. Otherwise, the alignment depends on the value of k: If $k \equiv 0 \mod 2^{\sigma'}$, then the matched filter is scale-aligned. We also distinguish between those symbols that began before the beginning of the matched filter interval — which are termed prior symbols — and those that begin after or at the same time — which are termed post symbols. The limits of summation for i in (74) are obtained by determination of how many signals in the summation are prior to the matched filter interval starting at kT and how many are post to the matched filter interval.

Consider first the number of prior and post symbols overlapping the matched filter when it is scale-aligned. For a matched filter of length q_sT, there are $q-1$ prior symbols on scale σ intersecting the same time as the scale s signal. The number of post symbols u on scale σ that overlap the matched filter is such that

$$q_s > (u-1)2^\sigma,$$

where $u-1$ represents the number of post symbols excluding the symbol with the same starting time as the matched filter. Since all quantities are integers, this is equivalent to

$$q_s - 1 \geq (u-1)2^\sigma.$$

The total number of post symbols, including the one with the same starting time, is therefore

$$u = \left\lfloor \frac{q_s - 1}{2^\sigma} \right\rfloor + 1 = \left\lfloor \frac{q2^s + 2^\sigma - 1}{2^\sigma} \right\rfloor, \tag{75}$$

where $\lfloor x \rfloor$ is the greatest integer not greater than x.

When the matched filter is non-scale-aligned, there are q prior overlapping symbols. The number of post symbols is determined as follows. Let $i \in 2^s\mathbb{Z}$ be a number between 0 and 2^σ representing the starting index of the matched filter on scale σ relative to the starting time of the nearest prior symbol. The number of symbols u on scale σ that overlap the symbol of length q_s starting at i is such that

$$q_s + i > u2^\sigma$$

or

$$q_s + i - 1 \geq u2^\sigma$$

so that

$$u = \left\lfloor \frac{q2^s + i - 1}{2^\sigma} \right\rfloor.$$

In the case that i is not in the range from 0 to 2^σ, this may be written as

$$u = \left\lfloor \frac{q2^s + (i \bmod 2^\sigma) - 1}{2^\sigma} \right\rfloor. \tag{76}$$

In (74) the index of summation must be changed so that the index $i = 0$ corresponds to the first post signal. To this end, we introduce the notation $(k)_s$ to represent the smallest integer $\geq k$ that is a multiple of 2^s. For example,

$$(k)_0 = k$$

$$(k)_1 = \begin{cases} k & \text{if } k \text{ is even} \\ k+1 & \text{if } k \text{ is odd} \end{cases}$$

$$(k)_2 = \begin{cases} k & \text{if } k \equiv 0 \bmod 4 \\ k+3 & \text{if } k \equiv 1 \bmod 4 \\ k+2 & \text{if } k \equiv 2 \bmod 4 \\ k+1 & \text{if } k \equiv 3 \bmod 4. \end{cases}$$

In (74), make the substitution $2^{\sigma'}i \to 2^{\sigma'}i + (k)_{\sigma'}$ and write it as

$$\begin{aligned} r_{s,1}(k) &= \sum_{\sigma=1}^{N_s+1} \sum_i b_{\sigma,(k)_{\sigma'}+i2^{\sigma'}} \int_{kT}^{(k+q_\sigma)T} p_s(t-kT)\xi(t)p_\sigma(t-(k)_{\sigma'}T - t_D - iT_{\sigma'}) \\ &\quad \exp[j2\pi(f_D t - (f_c+f_D)t_D)]dt \\ &= \sum_{\sigma=1}^{N_s+1} \sum_i b_{\sigma,(k)_{\sigma'}+i2^{\sigma'}} \int_0^{q_s T} p_s(t)\xi(t+kT)p_\sigma(t+(k-(k)_{\sigma'})T - iT_{\sigma'} - t_D) \\ &\quad \exp[j2\pi(f_D(t+kT) - (f_c+f_D)t_D)]dt. \end{aligned} \tag{77}$$

Observe that

$$k-(k)_\sigma = \begin{cases} 0 & \text{if } k \equiv 0 \bmod 2^\sigma \\ -(2^\sigma - (k \bmod 2^\sigma)) & k \not\equiv 0 \bmod 2^\sigma \end{cases}$$

so that the effect of the transformation is to shift the p_σ signal in (77) so that when $i = 0$ the p_σ signal starts at the same time or after the signal p_s; so p_σ is a post signal for $i \geq 0$. The index of summation in (77) can now be changed to reflect the number of prior and post signals:

$$\begin{aligned} r_{s,1}(k) = \sum_{\sigma=1}^{N_s+1} \sum_{i=l_k(\sigma')}^{u_{s',k}(\sigma')-1} & b_{\sigma,(k)_{\sigma'}+i2^{\sigma'}} \int_0^{q_sT} p_s(t)\xi(t+kT)p_\sigma(t+(k-(k)_{\sigma'})T - iT_{\sigma'} - t_D) \\ & \exp[j2\pi(f_D(t+kT) - (f_c + f_D)t_D)]dt \end{aligned} \tag{78}$$

where

$$l_k(\sigma) = \begin{cases} -(q-1) & k \equiv 0 \bmod 2^\sigma \text{ (scale-aligned)} \\ -q & k \not\equiv 0 \bmod 2^\sigma \text{ (non-scale-aligned)} \end{cases}$$

and, from (75) and (76),

$$u_{s,k}(\sigma) = \begin{cases} \left\lfloor \frac{2^s q + 2^\sigma - 1}{2^\sigma} \right\rfloor & k \equiv 0 \bmod 2^\sigma \text{ (scale-aligned)} \\ \left\lfloor \frac{2^s q + (k \bmod 2^\sigma) - 1}{2^\sigma} \right\rfloor & k \not\equiv 0 \bmod 2^\sigma \text{ (non-scale-aligned)} \end{cases}$$

It is straightforward to show that $r_{s,2}(k) = r_{s,1}(k - 2^{s'})$.

The autocorrelations are computed conditioned upon a known set of interfering bits, $\mathcal{I}_{s,k}$, which is the set of all bits referenced in (78). The correlation between fading components can now be written as

$$\begin{aligned} r_s(k;1,1|\mathcal{I}_{s,k}) &= \frac{1}{2}E[r^*_{s,1}(k)r_{s,1}(k)|\mathcal{I}_{s,k}] \\ &= \sum_{\sigma_1=1}^{N_s+1}\sum_{\sigma_2=1}^{N_s+1}\sum_{i_1=l_k(\sigma_1')}^{u_{s',k}(\sigma_1')-1}\sum_{i_2=l_k(\sigma_2')}^{u_{s',k}(\sigma_2')-1} b_{\sigma_1,(k)_{\sigma_1'}+i_1 2^{\sigma_1'}} b_{\sigma_2,(k)_{\sigma_2'}+i_2 2^{\sigma_2'}} \\ &\int_0^{q_sT}\int_0^{q_sT} p^*_s(t_1)p_s(t_2)R_\xi(t_2-t_1)p^*_{\sigma_1}(t_1+(k-(k)_{\sigma_1'})T - i_1T_{\sigma_1'} - t_D) \\ &p_{\sigma_2}(t_2+(k-(k)_{\sigma_2'})T - i_2T_{\sigma_2'} - t_D)\exp[2j\pi f_D(t_2-t_1)]dt_1dt_2. \end{aligned} \tag{79}$$

The correlation between fading components at lag one can be written as

$$\begin{aligned} r_s(k;1,2|\mathcal{I}_{s,k}) = r^*_s(k;2,1|\mathcal{I}_{s,k}) &= \frac{1}{2}E[r^*_{s,1}(k)r_{s,2}(k)|\mathcal{I}_{s,k}] \\ &= \sum_{\sigma_1=1}^{N_s+1}\sum_{\sigma_2=1}^{N_s+1}\sum_{i_1=l_k(\sigma_1')}^{u_{s',k}(\sigma_1')-1}\sum_{i_2=l_{(k-2^{s'})}(\sigma_2')}^{u_{s',(k-2^{s'})}(\sigma_2')-1} b_{\sigma_1,(k)_{\sigma_1'}+i_1 2^{\sigma_1'}} b_{\sigma_2,(k-2^{s'})_{\sigma_2'}+i_2 2^{\sigma_2'}} \\ &\int_0^{q_sT}\int_0^{q_sT} p^*_s(t_1)p_s(t_2)R_\xi(t_2-t_1-2^{s'}T)p^*_{\sigma_1}(t_1+(k-(k)_{\sigma_1'})T - i_1T_{\sigma_1'} - t_D) \\ &p_{\sigma_2}(t_2+((k-2^{s'})-(k-2^{s'})_{\sigma_2'})T - i_2T_{\sigma_2'} - t_D) \\ &\exp[2j\pi f_D(t_2-t_1-2^{s'}T)]dt_1dt_2 \end{aligned} \tag{80}$$

Appendix B Programs Used to Evaluate the Probability of Error for Fast Rician Channel of Section 3

The probability of error, presented by equation (19), when applying multiscale wavelet signaling is evaluated by `R11_12.c` and `comp_pe_Bit.c` which appear in algorithm 1. Due to the tremendous computation involved,

many tricks are used to speed up the computation. First, `R11_12.c` is used to calculate (79) and (80) without specifying $\mathcal{I}_{s,k}$. The result from `R11_12.c` is stored in a table corresponding to slot of different shifts of overlapping that will be imported to `comp_pe_Bit.c`. The program `comp_pe_Bit.c` will take the table output by `R11_12.c`, assigns different set of $\mathcal{I}_{s,k}$ and completes the calculation. As the size of $\mathcal{I}_{s,k}$ can be huge, it is impractical to calculate the probability of error for all the different combinations in $\mathcal{I}_{s,k}$. So, what `comp_pe_Bit.c` really does is randomly generate a large enough number of elements of $\mathcal{I}_{s,k}$ and use the average probability of error obtained from this subset of $\mathcal{I}_{s,k}$ to approximate the true probability of error.

Algorithm 1 Compute probability of error for Rician fading

```
/*************************************
*
*  Program: R11_12.c -- modified from Dr Moon's mason2.c
*
*  Chat Lo
*  Utah State University
*
*  Date:  Oct 14, 1998
*
*  This program computes the autocorrelations upon a known set
*  of interfering bits and output to a table that will be used
*  by comp_pe_Bit.c which will assign to each slot of the table
*  a relative data value (-1 or 1) to caculate the probability
*  of error by randomly generate a large number of trials.
*
*************************************/

#include <pmatlib.h>
#include <alib.h>
#include <usadtypes.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>


#define calloc_array pcalloc_array
#define calloc_matrix pcalloc_matrix

#define min(a,b) ((a) < (b) ? (a) : (b))
#define max(a,b) ((a) < (b) ? (b) : (a))
#define EPS 1.0e-10
#define DATALEN_N 1000

/* global variables */
int Q,                          /* length of wavelet signal */
double T0,                      /* intersymbol time for basic scale, when T0=1,
                                   intersymbol time for scale 1 will be 2 */
int Ns,                         /* number of scales, Ns */
int N_psi_phi,                  /* number of points for psidata/phidata */
char wavename[20],
int numcoef,                    /* number of coeff for psi/phi */
double psi_phi_samp_sd_t;       /* sampled time for psi/phi with T=1 */
double *psidata,                /* wavelet function */
double *phidata,                /* scaling function */
char outmap[80],
char outfile[80],
FILE *mapfp,
FILE *outfp,


int R_xitab.esize = 10000,
double *R_xitab.e,              /* fading correlation values  */
/**For R_xi tab.e on.y **/
double R_xi_sca.e,
/*************************/


double bessj0(double),
void init_R_xitab.e(double),    /* initiate R_xi table */
double R_xi(double),            /* return value for R_xi, fading correlation */
int .ower(int, int),            /* \(i_k(segma)\) */
int upper(int, int, int),       /* \(s_{s,k}(segma)\) */
int k_minus_ksig(int,int),      /* \((k-(k)_\segma)\) */
void init_psi_phi(char *),
/*** wavelet functions ***/
double psi(double),
double phi(double),
double psi1(double);            /* They are exactly the same, */
double phi1(double);            /* duplicated to speed up integration */
double psi2(double);
double phi2(double);
double scale_psi;
double scale_psi1;
double scale_psi2;
double scale_phi;
double scale_phi1;
double scale_phi2;

/*************************/


int prime(int);                 /* segma'=(segma<Ns?segma:Ns) */
double T_segma(int);            /* T_{segma}=2^{segma}T */
double deltat;

double qgaus(double (*func)(double), double, double);
```

```
void gauleg(void);
double x_pos[GAULEG_N], weg[GAULEG_N];
/****************************************
These global variables are for the multi integration
****************************************/
double A1,A2,B1,B2,C1,C2,T1,T2,T3;
int in_s, in_segma1, in_segma2;
double savt;
double (* P_s) (double);
double (* P_segma1) (double);
double (* P_segma2) (double);
double fint111(double);
double fint112(double);
double fint121(double);
double fint122(double);
int BigT;
double r11comp(int,int,int,int,int,int);/* r11comp(k,s,segma1,segma2,i1,i2) */
double r12comp(int,int,int,int,int,int);/* r12comp(k,s,segma1,segma2,i1,i2) */

/*************************************/


int main(int argc, char **argv){

  char *psifname;
  int psinamechar,type;
  int s,k,segma1,segma2,i1,i2;
  double Bd;
  int count=0;
  char fileename[20];
  int AdjDeltat;
  int i;


  if(argc == 1) {
    printf("usage: R11_12.c paramfilename\n");
    exit(0);
  }

  psifname = (char *)avread(argv[1],"psifname",&psinamechar,CHAR);
  asread argv[1],"TO",&TO,DOUBLE);
  asread argv[1],"Ns",&Ns,INT);
  asread argv[1],"Bd",&Bd,DOUBLE);
  asread argv[1],"type",&type,INT);
  asread argv[1],"s",&s,INT);
  asread argv[1],"Deltat",&deltat,DOUBLE);
  asread argv[1],"AdjDeltat",&AdjDeltat,INT);

  /*initia.ize abscissas and weights of the Gauss-Legendre quadrature formula*/
  gau.eg();

  if(AdjDe.tat)
    de.tat *= prime(s)/2.0;
  Big"s",segma prime s));



  ielt_psi_pai psifname);
  ielt_R_ritab.e Bd);
  sprintf("f1.eename,"%sdXg",argv[1],Bd);
  strcpy(outf1.e,f1.eename);
  strcat outf1.e,".out");
  strcpy outmap,"f1.eename);
  strcat outmap,".map");

  mapfp = fopen outmap,"w");
  shi.e 'mapfp'
    mapfp = fopen outmap,"w");
    printf("Can.d not open mapf1.e\n");
  }
  fprint" mapfp, "Is, "O=Lg, Ns=Ld, s=%d\n",savename,TO,Ns,s);
  fc.ose mapfp

  outfp = fopen outf1.e,"w");
  shi.e('outfp'
    outfp = fopen outf1.e,"w");
    printf("Can.d not open outf1.e\n");
  }
  fc.ose(outfp);

  if(type==1)
    for(i=-1 i( i<<(Ns*prime(s)))),i++){
      k=i.<<prime's);
      mapfp = fopen outmap,"a");
      shi.e 'mapfp'
mapfp = fopen outmap,"a");
printf("Can.d not open mapf1.e\n");
      }
      fprintf(mapfp,"k=%d\t",k);
      fc.ose(mapfp);
      for(segma1=1,segma1<=(Ns+1);segma1++){
mapfp = fopen outmap,"a");
shi.e('mapfp){
  mapfp = fopen.outmap,"a");
  printf("Can.d not open mapf1:e\n");
}
fprintf(mapfp,"segma1=%d\t",segma1);
fclose(mapfp);
for(segma2=1;segma2<=(Ns+1);segma2++){
  mapfp = fopen(outmap,"a");
  shi.e('mapfp){
    mapfp = fopen(outmap,"a");
    printf("Can.d not open mapf1:e\n");
  }
  fprintf(mapfp,"segma2=%d\t",segma2);
  fclose(mapfp);
```

```
      for(i1=lower(prime(segma1),k);i1<upper(prime(segma1),prime(s),k);i1++){
        mapfp = fopen(outmap,"a");
        while(!mapfp){
          mapfp = fopen(outmap,"a");
          printf("Could not open mapfile\n");
        }
      fprintf(mapfp,"i1=%d\n",i1);
        fclose(mapfp);
        for(i2=lower(prime(segma2),k);i2<upper(prime(segma2),prime(s),k);i2++){
          mapfp = fopen(outmap,"a");
          while(!mapfp){
    mapfp = fopen(outmap,"a");
    printf("Could not open mapfile\n");
          }
          fprintf(mapfp,"i2=%d :",i2);
          fclose(mapfp);
          printf("%g\n",r11comp(k,s,segma1,segma2,i1,i2));
          printf("count=%d \n",++count);
        }
        mapfp = fopen(outmap,"a");
        while(!mapfp){
          mapfp = fopen(outmap,"a");
          printf("Could not open mapfile\n");
        }
        fprintf(mapfp,"\n");
        fclose(mapfp);
        outfp = fopen(outfile,"a");
        while(!outfp){
          outfp = fopen(outfile,"a");
          printf("Could not open outfile\n");
        }
        fprintf(outfp,"\n");
        fclose(outfp);
      }
    }
          }
        }
      }
      else{
        for(i=0,i<(1<<(%s-prime(s)));i++){
          k=i+(1<<prime(s));
          mapfp = fopen(outmap,"a");
          while(!mapfp){
    mapfp = fopen(outmap,"a");
    printf("Could not open mapfile\n");
          }
          fprintf(mapfp,"k=%d\t",k);
          fclose(mapfp);
          for(segma1=1;segma1<=(%s+1);segma1++){
    mapfp = fopen(outmap,"a");
    while(!mapfp){
      mapfp = fopen(outmap,"a");
      printf("Could not open mapfile\n");
    }
    fprintf(mapfp,"segma1=%d\t",segma1);
    fclose(mapfp);
    for(segma2=1;segma2<=(%s+1);segma2++){
      mapfp = fopen(outmap,"a");
      while(!mapfp){
        mapfp = fopen(outmap,"a");
        printf("Could not open mapfile\n");
      }
      fprintf(mapfp,"segma2=%d\t",segma2);
      fclose(mapfp);
      for(i1= over(prime(segma1),k);i1<upper(prime(segma1),prime(s),k);i1++){
        mapfp = fopen(outmap,"a");
        while(!mapfp){
          mapfp = fopen(outmap,"a");
          printf("Could not open mapfile\n");
        }
        fprintf(mapfp,"i1=%d\n",i1);
        fclose(mapfp);
        for(i2=over(prime(segma2),(k-(1<<prime(s)))));i2<upper(prime(segma2),prime(s),(k-(1<<prime(s))));i2++){
          mapfp = fopen(outmap,"a");
          while(!mapfp){
    mapfp = fopen(outmap,"a");
    printf("Could not open mapfile\n");
          }
          fprintf(mapfp,"i2=%d  ",i2);
          fclose(mapfp);
          printf("%g\n",r12comp(k,s,segma1,segma2,i1,i2));
          printf("count=%d \n",++count);
        }
        mapfp = fopen(outmap,"a");
        while(!mapfp){
          mapfp = fopen(outmap,"a");
          printf("Could not open mapfile\n");
        }
        fprintf(mapfp,"\n");
        fclose(mapfp);
        outfp = fopen(outfile,"a");
        while(!outfp){
          outfp = fopen(outfile,"a");
          printf("Could not open outfile\n");
        }
        fprintf(outfp,"\n");
        fclose(outfp);
      }
    }
          }
        }
      }
      mapfp = fopen(outmap,"a");
      while(!mapfp){
        mapfp = fopen(outmap,"a");
        printf("Could not open mapfile\n");
      }
```

```
  fprintf(mapfp,"DONE!!!!!\n");
  fclose(mapfp);
}


void init_R_xitable(double Bd){
  /* Initiate the R_xi table. The table has R_xitablesize, and initiated
     according to the max tau required ( \(q_s T \) where s is the max
     scale), that is, R_xi[0] for R_xi(0), R_xi[R_xitablesize-1]=R_xi(maxtau).
  */
  int i;
  double tau,total_time;

  total_time = (double) (T0*(1<<Ms)*Q)+BigT;
  tau = total_time/(R_xitablesize-1);
  R_xitable = (double *)calloc_array(DOUBLE,R_xitablesize,"R_xitable");
  for(i = 0; i < R_xitablesize; i++) {
    R_xitable[i] = bessj0(2*M_PI*Bd*i*tau);
  }
  R_xi_scale = (R_xitablesize-1)/total_time;

}


double R_xi(double tau){
  /* Assume the function is linear between neighboring data

     R_xi(tau) = R_xitable[n] + delta*(R_xitable[n+1]-R_xitable[n])
     where n = int part of tau*scale, delta = fraction part of tau*scale.
  */

  int n;
  double delta,  scaled_time;


  tau =fabs(tau);
  scaled_time = (tau*R_xi_scale);
  n = (int) scaled_time;
  if(n==(R_xitablesize-1))
    return(R_xitable[n]);
  else if(n<(R_xitablesize-1)){
    delta = scaled_time - n;
    return(R_xitable[n] + delta*(R_xitable[n+1]-R_xitable[n]));
  }
  else{
    printf("R_xi out of range!n=%d tau=%g\n",n,tau);
    return 0;
  }
}


int lower(int sigma, int k)
{
   int s2;

   s2 = (1<<sigma);
   while(k < 0) k += s2;
   k = k%s2;
   if(!k) return -(Q-1);
   else return -Q;
}

int upper(int sigma, int n, int k)
{
   int tsigma;
   tsigma = (1<<sigma);
   while(k < 0) k += tsigma;
   k = k%tsigma;
   if(!k) {
  return (int)((Q<<n) + tsigma-1)/tsigma);
   }
   else {
  return (int)((Q<<n) + k - 1)/tsigma);
   }
}

int k_minus_ksig(int k,int sigma)
/* compute k - (k)_sigma */
{
   int s2;

   s2 = 1<<sigma;
   while(k < 0) k += s2;
   k = k % s2;
   if(!k) return 0;
   return(k - s2);
}

void init_psi_phi(char *fname){

  FILE *in_psi_phi_fi;
  int i;
  char str[100];


  /* psifile format:
     Name = short name of wavelet file
     N = number of coefficients (integer)
     psi_phi_sampled_t = sampled time (double)
     Q = length of wavelet (in integers) (integer)
     N_psi_phi = number of samples (integer)
     sample1, sample2, ... (double)  (wavelet)
     sample1, sample2, ... (double)  (scaling)
     */
  /* the data is made using the program "makewave" */

  in_psi_phi_file = fopen(filename,"r");
```

```
  if(in_psi_phi_file == NULL) {
    printf("Could not open in_psi_phi_file\n");
    exit(0);
  }
  fscanf(in_psi_phi_file,"%s\n",wavename);
  fgets(str,100,in_psi_phi_file); numcoef = atoi(str);
  fgets(str,100,in_psi_phi_file); psi_phi_sampled_t = atof(str);
  fgets(str,100,in_psi_phi_file); Q = atoi(str);
  fgets(str,100,in_psi_phi_file); N_psi_phi = atoi(str);
  psidata = (double *)calloc_array(DOUBLE,(N_psi_phi+1),"psidata");
  phidata = (double *)calloc_array(DOUBLE,(N_psi_phi+1),"phidata");
  for(i = 0; i < N_psi_phi; i++) {
    if(!fgets(str,100,in_psi_phi_file)) {
      printf("Cannot read sample\n");
      exit(0);
    }
    psidata[i] = atof(str);
  }
  psidata[N_psi_phi] = 0;
  for(i = 0; i < N_psi_phi; i++) {
    if(!fgets(str,100,in_psi_phi_file)) {
      printf("Cannot read sample\n");
      exit(0);
    }
    phidata[i] = atof(str);
  }
  phidata[N_psi_phi] = 0;
  fclose(in_psi_phi_file);
}


double psi(double t){
  /* Assume the function is linear between neighboring data

     psi(tau) = psidata[n] + delta*(psidata[n+1]-psidata[n])
     where n = int part of tau*scale, delta = fraction part of tau*scale.
  */
  int n;
  double delta, scaled_time;

  scaled_time = t/scale_psi;
  n = (int) scaled_time;
  if(n>=N_psi_phi)
    return(psidata[N_psi_phi]);
  else {
    delta = scaled_time - n;
    return(psidata[n] + delta*(psidata[n+1]-psidata[n]));
  }
}

double phi(double t){
  /* Assume the function is linear between neighboring data

     phi(tau) = phidata[n] + delta*(phidata[n+1]-phidata[n])
     where n = int part of tau*scale, delta = fraction part of tau*scale.
  */
  int n;
  double delta, scaled_time;

  scaled_time = t/scale_phi;
  n = (int) scaled_time;
  if(n>=N_psi_phi)
    return(phidata[N_psi_phi]);
  else {
    delta = scaled_time - n;
    return(phidata[n] + delta*(phidata[n+1]-phidata[n]));
  }
}

double psi1(double t){

  int n;
  double delta, scaled_time;

  scaled_time = t/scale_psi1;
  n = (int) scaled_time;
  if(n>=N_psi_phi)
    return(psidata[N_psi_phi]);
  else {
    delta = scaled_time - n;
    return(psidata[n] + delta*(psidata[n+1]-psidata[n]));
  }
}

double phi1(double t){

  int n;
  double delta, scaled_time;

  scaled_time = t/scale_phi1;
  n = (int) scaled_time;
  if(n>=N_psi_phi)
    return(phidata[N_psi_phi]);
  else {
    delta = scaled_time - n;
    return(phidata[n] + delta*(phidata[n+1]-phidata[n]));
  }
}

double psi2(double t){

  int n;
  double delta, scaled_time;

  scaled_time = t/scale_psi2;
  n = (int) scaled_time;
  if(n>=N_psi_phi)
```

```
    return(psidata[N_psi_phi]);
  else{
    delta = scaled_time - n;
    return(psidata[n] + delta*(psidata[n+1]-psidata[n]));
  }
}

double phi2(double t){

  int n;
  double delta, scaled_time;

  scaled_time = t/scale_phi2;
  n = (int) scaled_time;
  if(n>=N_psi_phi)
    return(phidata[N_psi_phi]);
  else{
    delta = scaled_time - n;
    return(phidata[n] + delta*(phidata[n+1]-phidata[n]));
  }
}

int prime(int sigma){
/* sigma'=(sigma<Ns?sigma:sigma-1) */

  if(sigma < Ns)
    return sigma;
  else
    return Ns;
}

double T_sigma(int sigma){
/* T_{sigma}=2^{sigma}T */

  return (T0*(1<<sigma));
}


double first22(double tau){
  double integral;

  integral = R_zi(tau-Big*);
  integral *=P_s((savt+tau));
  integral *=P_sigma2((savt+tau+T2));
  return integral;
}


double first21(double t){
  double integral;

  savt = t;
  integral = qgaus(first22,(B1-t),(B2-t));
  integral *=P_s(t);
  integral *=P_sigma1((t+T1));
  return integral;
}

double first12(double tau){
  double integral;

  integral = R_zi(tau);
  integral *=P_s((savt+tau));
  integral *=P_sigma2((savt+tau+T2));
  return integral;
}


double first11(double t){
  double integral;

  savt = t;
  integral = qgaus(first12,(B1-t),(B2-t));
  integral *=P_s(t);
  integral *=P_sigma1((t+T1));
  return integral;
}


double riicomp(int N,int s, int sigma1, int sigma2, int i1, int i2){
  /* riicomp(N,s,sigma1,sigma2,i1,i2) */
  double integral;
  double sca,sing_factor;

  if(s<=Ns){
    P_s = psi;
    is_s = s;
  }
  else{
    P_s = phi;
    is_s = Ns;
  }
  sca=s_psi = (double) (1<<is_s);
  sca=s_psi *=psi_phi_samp.sd_t;
  sca=s_phi = sca=s_psi;

  if(sigma1<=Ns){
    P_sigma1 = psi;
    is_sigma1 = sigma1;
  }
  else{
    P_sigma1 = phi;
    is_sigma1 = Ns;
  }
  scale_psi1 = (double) (1<<is_sigma1);
  scale_psi1 *=psi_phi_samp.sd_t;
  scale_phi1 = scale_psi1;
```

```
  if(sigma2<=Ns){
    P_sigma2 = psi2;
    in_sigma2 = sigma2;
  }
  else{
    P_sigma2 = phi2;
    in_sigma2 = Ns;
  }

  scale_psi2 = (double) (1<<in_sigma2);
  scale_psi2 *=psi_phi_sampled_t;
  scale_phi2 = scale_psi2;

  T1 = T0*k_minus_ksig(k,in_sigma1)-i1*T_sigma(in_sigma1);
  T2 = T0*k_minus_ksig(k,in_sigma2)-i2*T_sigma(in_sigma2);
  A1 = max((-T1),0);
  A2 = min((T0*(Q<<in_sigma1)-T1),(T0*(Q<<in_s)));
  B1 = max((-T2),0);
  B2 = min((T0*(Q<<in_sigma2)-T2),(T0*(Q<<in_s)));
  integral = qgaus(fint111, A1,A2);

  /* For used with simple()*/
  /*
  integral *=deltat;
  integral *=deltat;
  */
  /*
    This takes care the scaling of psi and phi which was not done
    in psi() and phi().
  */
  scaleing_factor = (double)(1<<(in_sigma1+in_sigma2));
  scaleing_factor = sqrt(scaleing_factor);
  scaleing_factor *= (double)(1<<in_s);
  integral /= scaleing_factor;

  mapfp = fopen(ostmap,"a");
  while(!mapfp){
    mapfp = fopen(ostmap,"a");
    printf("Could not open mapfile\n");
  }
  fprintf(mapfp,"%g ",integral);
  fclose(mapfp);
  outfp = fopen(outfile,"a");
  while(!outfp){
    outfp = fopen(outfile,"a");
    printf("Could not open outfile\n");
  }
  fprintf(outfp,"%g ",integral);
  fclose(outfp);
  return integral;
}

double r12comp(int k,int s, int sigma1, int sigma2, int i1, int i2){
  /* r12comp(k,s,sigma1,sigma2,i1,i2) */

  double integral;
  double scaleing_factor;

  if(s<=Ns){
    P_s = psi;
    in_s = s;
  }
  else{
    P_s = phi;
    in_s = Ns;
  }
  scale_psi = (double) (1<<in_s);
  scale_psi *=psi_phi_sampled_t;
  scale_phi = scale_psi;

  if(sigma1<=Ns){
    P_sigma1 = psi1;
    in_sigma1 = sigma1;
  }
  else{
    P_sigma1 = phi1;
    in_sigma1 = Ns;
  }
  scale_psi1 = (double) (1<<in_sigma1);
  scale_psi1 *=psi_phi_sampled_t;
  scale_phi1 = scale_psi1;

  if(sigma2<=Ns){
    P_sigma2 = psi2;
    in_sigma2 = sigma2;
  }
  else{
    P_sigma2 = phi2;
    in_sigma2 = Ns;
  }
  scale_psi2 = (double) (1<<in_sigma2);
  scale_psi2 *=psi_phi_sampled_t;
  scale_phi2 = scale_psi2;

  T1 = T0*k_minus_ksig(k,in_sigma1)-i1*T_sigma(in_sigma1);
  T2 = T0*k_minus_ksig((k-(1<<in_s)),in_sigma2)-i2*T_sigma(in_sigma2);
  A1 = max((-T1),0);
  A2 = min((T0*(Q<<in_sigma1)-T1),(T0*(Q<<in_s)));
  B1 = max((-T2),0);
  B2 = min((T0*(Q<<in_sigma2)-T2),(T0*(Q<<in_s)));
  integral = qgaus(fint121, A1,A2);
  scaleing_factor = (double)(1<<(in_sigma1+in_sigma2));
  scaleing_factor = sqrt(scaleing_factor);
  scaleing_factor *= (double)(1<<in_s);
  integral /= scaleing_factor;
```

```
  mapfp = fopen(outmap,"a");
  while(!mapfp){
    mapfp = fopen(outmap,"a");
    printf("Could not open mapfile\n");
  }
  fprintf(mapfp,"%g ",integral);
  fclose(mapfp);
  outfp = fopen(outfile,"a");
  while(!outfp){
    outfp = fopen(outfile,"a");
    printf("Could not open outfile\n");
  }
  fprintf(outfp,"%g ",integral);
  fclose(outfp);
  return integral;
}


/*************************************************
*
* Numerical integration
*
*************************************************/

void gauleg(void)
{

  int j,i;
  double z1, z, xi, pp, p3, p2, p1;

  z1 = 1;

  for(i=0;i<GAULEG_N;i++){
    z = cos(M_PI*(i+0.75)/(2*GAULEG_N+0.5));
    do{
      p1=1.0;
      p2=0.0;
      for(j=0;j<(2*GAULEG_N);j++){
p3=p2;
p2=p1;
p1=((2.0*j+1)*z*p2-j*p3)/(j+1);
      }
      pp=2*GAULEG_N*(z*p1-p2)/(z*z-1);
      z1=z;
      z=z1-p1/pp;
    }while(fabs(z-z1)>EPS);
    x_pos[GAULEG_N-1-i]= xi*z;
    weg[GAULEG_N-1-i]=2.0*xi/((1.0-z*z)*pp*pp);
  }
}

doub.e qgaus(doub.e (*func)(doub.e), double a, double b)
{
  int j;
  doub.e xr, xm, dx, s;


  xm=0.5* b+a);
  xr=0.5* b-a);
  s=0;
  for(j=0;j<GAULEG_N;j++){
    dx=xr*x_pos[j];
    s += weg[j]* ((*func)(xm+dx)+(*func)(xm-dx));
  }
  return s *xr
}

doub.e beesj0 doub.e x;
{
        doub.e ax,z;
        doub e rx,y,ans,ans1,ans2;

        if ((ax=fabs(x)) < 8.0) {
                y=x*x;
                ans1=57568490574.0+y*(-13362590354.0+y*(651619640.7
                        +y*(-11214424.18+y*(77392.33017+y*(-184.9052456)))));
                ans2=57568490411.0+y*(1029532985.0+y*(9494680.718
                        +y*(59272.64853+y*(267.8532712+y*1.0))));
                ans=ans1/ans2;
        } else {
                z=8.0/ax;
                y=z*z;
                xx=ax-0.785398164;
                ans1=1.0+y*(-0.1098628627e-2+y*(0.2734510407e-4
                        +y*(-0.2073370639e-5+y*0.2093887211e-6)));
                ans2 = -0.1562499995e-1+y*(0.1430488765e-3
                        +y*(-0.6911147651e-5+y*(0.7621095161e-6
                        -y*0.934935152e-7)));
                ans=sqrt(0.636619772/ax)*(cos(xx)*ans1-z*sin(xx)*ans2);
        }
        return ans;
}


/*
Local Variables:
compile-command:"apcc -o R11_12 -g R11_12.c -I/usr/include/local  -lpmatlib -lim -lm"
;compile-command:"gcc -o R11_12 -g R11_12.c -I/usr/include/local -L/opt/lib  -lpmatlib -lim -lm"
;compile-command:"gcc -o dR11_12 -g R11_12.c -I/usr/include/local -L/opt/lib  -lpmatlib -lim -lm"
End:
*/
```

```
/*****************************************
*
*  Program: comp_pe_Bit.c -- modified from Dr Moon's mason2.c
*
*  Chet Lo
*  Utah State University
*
*  Date:  Oct 14, 1998
*
*****************************************/

#include <pmatlib.h>
#include <alib.h>
#include <asadtypes.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>


#define calloc_array pcalloc_array
#define calloc_matrix pcalloc_matrix

#define min(a,b) ((a) < (b) ? (a) : (b))
#define max(a,b) ((a) < (b) ? (b) : (a))
#define EPS 1.0e-10
#define GAULEG_N 1000


/* global variables */
int Q;                          /* length of wavelet signal */
double T0;                      /* intersymbol time for basic scale, when T0=1,
   intersymbol time for scale 1 will be 2 */
double N0,
double Nd,
double S,
double D,
int Ns,                         /* number of scales, Ns */

int **lower(int, int);          /* \(l_k(segma)\) */
int **upper(int, int, int);     /* \(u_{s,k}(segma)\) */
int k_minus_ksig(int,int);      /* \((k-(k)_\segma)\) */
int **ksig(int, int);
int *prime(int);                /* segma'=(segma<Ns?segma:Ns) */
double *T_segma(int);           /* T_{segma}=2^{segma}T */

double *******r11;              /* r11[s][i][segma1][segma2][i1][i2] */
double *******r12;

void init_r11(char *);
void init_r11(char *);

double C,V,W,psiang,e,eta,De,taPhi;
double P(double,int,int);
double fiat_double();
double de,tst = 1e-3,

double qgaus(double (*func)(double), double, double);
void gauleg(void);
double x_pos[GAULEG_N], weg[GAULEG_N];



double r11_val_se(int,int);
double r12_val_se(int,int);
double r22_val_se(int,int);
double comp_Pe(void);
double comp_De_taPhi(int, int);
void setbitmap(void);           /* randomly set the bitmap table */

/** Bit stuff **/
int **bitmap;                   /* bit patterns */
int *max_position, *min_position;  /* for cal the dimension of bitmap */
void init_bitmap_void();


void init_bitmap(void){

  int s,i,k,segma1,segma2,i1,i2,
  int temp1, temp2;
  /* Used to store the max/min position for individual scales */
  int j;


  max_position = (int *)malloc(sizeof(int)*(Ns+1))-1;
  min_position = (int *)malloc(sizeof(int)*(Ns+1))-1;

  for(j=1;j<=(Ns+1);j++){
    max_position[j] = 0;
    min_position[j] = 0;
  }

  for(s=1;s<=(Ns+1);s++){
    for(i=-1;i<(1<<(Ns-prime(s)));i++){
      k=i*(1<<prime(s));
      for(segma1=1;segma1<=(Ns+1);segma1++){
for(segma2=1;segma2<=(Ns+1);segma2++){
  for(i1=lower(prime(segma1),k);i1<upper(prime(segma1),prime(s),k);i1++){
    for(i2=lower(prime(segma2),k);i2<upper(prime(segma2),prime(s),k);i2++){
      temp1=(ksig(k,prime(segma1))+i1*(1<<prime(segma1)))/(1<<prime(segma1));
      temp2=(ksig(k,prime(segma2))+i2*(1<<prime(segma2)))/(1<<prime(segma2));
      if(temp1<min_position[segma1])
```

```
min_position[segma1]=temp1;
      else if(temp1>max_position[segma1])
max_position[segma1]=temp1;
      if(temp2<min_position[segma2])
min_position[segma2]=temp2;
      else if(temp2>max_position[segma2])
max_position[segma2]=temp2;
    }
  }
}
      }
    }
  }

  for(s=1;s<=(Ns+1);s++){
    for(i=0;i<(1<<(Ns-prime(s)));i++){
      k=i*(1<<prime(s));
      for(segma1=1;segma1<=(Ns+1);segma1++){
for(segma2=1;segma2<=(Ns+1);segma2++){
  for(i1=lower(prime(segma1),k);i1<upper(prime(segma1),prime(s),k);i1++){
    for(i2=lower(prime(segma2),(k-(i1<<prime(s))));i2<upper(prime(segma2),prime(s),(k-(i1<<prime(s))));i2++){
      temp1=(ksig(k,prime(segma1))+i1*(1<<prime(segma1)))/(1<<prime(segma1));
      temp2=(ksig((k-(i1<<prime(s))),prime(segma2))+i2*(1<<prime(segma2)))/(1<<prime(segma2));
      if(temp1<min_position[segma1])
min_position[segma1]=temp1;
      else if(temp1>max_position[segma1])
max_position[segma1]=temp1;
      if(temp2<min_position[segma2])
min_position[segma2]=temp2
      else if(temp2>max_position[segma2])
max_position[segma2]=temp2;
    }
  }
}
      }
    }
  }
  for(s=1;s<=(Ns+1);s++){
    printf("%d %d\n",min_position[s],max_position[s]);
  }

  bitmap = (int **)malloc(sizeof(int *)*(Ns+1))-1;
  for(s=1;s<=(Ns+1);s++){
    bitmap[s]=(int*)malloc(sizeof(int)*(max_position[s]-min_position[s]+1))-min_position[s];
  }
}


int lower(int segma, int k)
{
   int s2;

   s2 = (1<<segma);
   while(k < 0) k += s2;
   k = k%s2;
   if(!k) return -(Q-1);
   else return -Q;
}

int upper(int segma, int s, int k)
{
   int tsegma;
   tsegma = (1<<segma);
   while(k < 0) k += tsegma;
   k = k%tsegma;
   if(!k) {
  return (int)((Q<<s) + tsegma-1)/tsegma);
   }
   else {
  return (int)((Q<<s) + k - 1)/tsegma);
   }
}

int k_minus_ksig(int k,int segma)
/* compute k - (k)_segma */
{
   int s2;

   s2 = 1<<segma;
   while(k < 0) k += s2;
   k = k % s2;
   if(!k) return 0;
   return(k - s2);
}

int ksig(int k, int segma){
  int temp;

  temp = k_minus_ksig(k,segma);
  temp -= k;
  return (-temp);
}

int prime(int segma){
/* segma'=(segma<Ns?segma:segma-1) */

  return(min(segma,Ns));

}

double T_segma(int segma){
/* T_(segma)=2^(segma)T */

  return (T0*(1<<segma));
}

void init_rll(char *name){
```

```
  int s,k,sigma1,sigma2,i1,i2;
  int i;
  char filename[30];
  FILE *infile;

  r11 =(double******)malloc(sizeof(double*****)*(Ns+1))-1;
  for(s=1;s<=(Ns+1);s++){
    r11[s] =(double*****)malloc(sizeof(double****)*((1<<(Ns-prime(s)))+1))+1;
    /* +1 so that the smallest value for i can be -1 */
    for(i=-1;i<(1<<(Ns-prime(s)));i++){
      k=i*(1<<prime(s));
      r11[s][i]=(double****)malloc(sizeof(double***)*(Ns+1))-1;
      for(sigma1=1;sigma1<=(Ns+1);sigma1++){
  r11[s][i][sigma1]=(double***)malloc(sizeof(double**)*(Ns+1))-1;
for(sigma2=1;sigma2<=(Ns+1);sigma2++){
  r11[s][i][sigma1][sigma2]=(double**)malloc(sizeof(double*)*(upper(prime(sigma1),prime(s),k)-lower(prime(sigma1),k)))-lower(prime(sigma1),k);
  for(i1=lower(prime(sigma1),k);i1<upper(prime(sigma1),prime(s),k);i1++){
    r11[s][i][sigma1][sigma2][i1]=(double*)malloc(sizeof(double)*(upper(prime(sigma2),prime(s),k)-lower(prime(sigma2),k)))-lower(prime(sigma2),k);
  }
}
      }
    }
  }

  for(s=1;s<=(Ns+1);s++){
    sprintf(filename,"%sKdBdKg",name,s,Bd);
    strcat(filename,".out");
    infile=fopen(filename,"r");
    if(infile==NULL){
      printf("Not able to open %s\n",filename);
      exit(0);
    }
    for(i=-1;i<(1<<(Ns-prime(s)));i++){
      k=i*(1<<prime(s));
      for(sigma1=1;sigma1<=(Ns+1);sigma1++){
for(sigma2=1;sigma2<=(Ns+1);sigma2++){
  for(i1=lower(prime(sigma1),k);i1<upper(prime(sigma1),prime(s),k);i1++){
    for(i2=lower(prime(sigma2),k);i2<upper(prime(sigma2),prime(s),k);i2++){
       fscanf(infile,"%le",&r11[s][i][sigma1][sigma2][i1][i2]);
    }
  }
}
      }
    }
    fclose(infile);
  }
}

void init_ri2(char *name){

  int s,k,sigma1,sigma2,i1,i2;
  int i;
  char fi.ename[30];
  FILE *infi.e;

  ri2 =(doub.e******)malloc(sizeof(double*****)*(Ns+1))-1;
  for(s=1;s<=(Ns+1);s++){
    ri2[s] =(doub.e*****)malloc(sizeof(double****)*(1<<(Ns-prime(s))));
    for(i=0;i<(1<<(Ns-prime(s)));i++){
      k=i*(1<<prime(s));
      ri2[s][i]=(doub.e****)malloc(sizeof(double***)*(Ns+1))-1;
      for(sigma1=1;sigma1<=(Ns+1);sigma1++){
  ri2[s][i][sigma1]=(doub.e***)malloc(sizeof(double**)*(Ns+1))-1;
for(sigma2=1;sigma2<=(Ns+1);sigma2++){
  ri2[s][i][sigma1][sigma2]=(doub.e**)malloc(sizeof(double*)*(upper(prime(sigma1),prime(s),k)-lower(prime(sigma1),k)))-lower(prime(sigma1),k);
  for(i1=.ower(prime(sigma1),k);i1<upper(prime(sigma1),prime(s),k);i1++){
    ri2[s][i][sigma1][sigma2][i1]=(doub.e*)malloc(sizeof(double)*(upper(prime(sigma2),prime(s),(k-(1<<prime(s))))-lower(prime(sigma2),(k-(1<<prime(s))))))-lower(prime(sigma2),(k-(1<<prime(s))));
  }
}
      }
    }
  }

  for(s=1;s<=(Ns+1);s++){
    sprintf(fi.ename,"%sKdBdKg",name,s,Bd);
    strcat(fi.ename,".out");
    infi.e=fopen(fi.ename,"r");
    if(infi.e==NULL){
      printf("Not ab.e to open %s\n",fi.ename);
      exit(0);
    }
    for(i=0;i<(1<<(Ns-prime(s)));i++){
      k=i*(1<<prime(s));
      for(sigma1=1;sigma1<=(Ns+1);sigma1++){
for(sigma2=1;sigma2<=(Ns+1);sigma2++){
  for(i1=.ower(prime(sigma1),k);i1<upper(prime(sigma1),prime(s),k);i1++){
    for(i2=lower(prime(sigma2),(k-(1<<prime(s))));i2<upper(prime(sigma2),prime(s),(k-(1<<prime(s))));i2++){
       fscanf(infi.e,"%.e",&ri2[s][i][sigma1][sigma2][i1][i2]);
    }
  }
}
      }
    }
    fclose(infi.e);
  }
}


double ri1_value(int s, int i){
  int sigma1, sigma2, i1,i2,k;
  double temp, sum=0;


  k=i*(1<<prime(s));
  for(sigma1=1;sigma1<=(Ns+1);sigma1++){
    for(sigma2=1;sigma2<=(Ns+1);sigma2++){
```

```
      for(i1=lower(prime(segma1),k);i1<upper(prime(segma1),prime(s),k);i1++){
for(i2=lower(prime(segma2),k);i2<upper(prime(segma2),prime(s),k);i2++){
  temp = r11[s][i][segma1][segma2][i1][i2];
  temp *=bitmap[segma1][(ksig(k,prime(segma1))+i1*(1<<prime(segma1)))/(1<<prime(segma1))];
  temp *=bitmap[segma2][(ksig(k,prime(segma2))+i2*(1<<prime(segma2)))/(1<<prime(segma2))];
  sum += temp;
}
      }
    }
  }
  sum *=D;
  return sum;
}

double r12_value(int s, int i){
  int segma1, segma2, i1,i2,k;
  double temp, sum=0;

  k=i*(1<<prime(s));
  for(segma1=1;segma1<=(Ns+1);segma1++){
    for(segma2=1;segma2<=(Ns+1);segma2++){
      for(i1=lower(prime(segma1),k);i1<upper(prime(segma1),prime(s),k);i1++){
for(i2=lower(prime(segma2),(k-(1<<prime(s))));i2<upper(prime(segma2),prime(s),(k-(1<<prime(s))));i2++){
  temp = r12[s][i][segma1][segma2][i1][i2];
  temp *=bitmap[segma1][(ksig(k,prime(segma1))+i1*(1<<prime(segma1)))/(1<<prime(segma1))];
  temp *=bitmap[segma2][(ksig((k-(1<<prime(s))),prime(segma2))+i2*(1<<prime(segma2)))/(1<<prime(segma2))];
  sum += temp;
}
      }
    }
  }
  sum *=D;
  return sum;
}

doub.e r22_value(int s, int k){
  return r11_value(s,(k-1));
}



doub.e P(doub.e localpsi, int s, int k)
{
  doub.e k11,k22,k12;
  doub.e rho1, rho2;
  doub.e integral;

  k11 = r11_va.ue(s,k)+H0;
  k12 = r12_value(s,k);
  k22 = H0+r22_value(s,k);

  if(k11<0)
    k11 = -k11;

  if(k22<0)
    k22 = -k22;


  rho1 = 5/k11.
  rho2 = 5/k22.
  eta = k12/.sqrt(k11)*sqrt(k22));
  b =  b* rho1*rho2;.
  W =  a* rho2-rho1.;
  M = sqrt rho1*rho2;.
  psiang = .oca.psi.

  integra. = qgaus flat, -M_PI/2, M_PI/2)/(4*M_PI);

  return integra..
}


doub.e flat.doub.e t.
{
  doub.e E.
  doub.e ti, resu.t
  /* assumes that De.taPhi = 0 and psiangle = +-pi/2 */
  /* for De.taPhi = +-pi, psiang.e=+-pi/2, cos(DeltaPhi-psiangle)=0*/
  E = (1 - V*sin(t));.
  if(E==0
    print*("E=0\n");.
  ti = W*sin(De.taPhi-psiang.e)/E;
  ti += (eta*sin(psiang.e));.
  resu.t = exp(-E*ti.
  return(resu.t);.

}

doub.e comp_De.taPhi(int s, int k){
  int temp;

  temp = bitmap[s][k]-bitmap[s][k-1];
  if(temp==0)
    return 0.
  e.se
    return M_PI;
}

double comp_Pe(void){
  int s,k;
  double Pei, Pe=0, temp;

  for(s=1;s<=(Ns+1);s++){
    Pei=0;
    for(k=0;k<(1<<(Ns-prime(s)));k++){
      DeltaPhi=comp_DeltaPhi(s,k);
```

```
      temp = F((M_PI/2),s,k) - F((-M_PI/2),s,k);
      Pai+=fabs(temp);
    }
    Pai /= (double)(1<<(Ns-prime(s)));
    Pai /= (double)(1<<prime(s));
    Pe += Pai;
  }

  return Pe;
}


void setbitmap(void){
  int s,k;

  for(s=1;s<=(Ns+1);s++){
    for(k=min_position[s];k<=max_position[s];k++){
      if(rand()<(RAND_MAX/2))
bitmap[s][k] = 1;
      else
bitmap[s][k] = -1;
    }
  }
  printf("\n");
  for(s=1;s<=(Ns+1);s++){
    for(k=min_position[s];k<=max_position[s];k++){
      printf("%d ",bitmap[s][k]);
    }
  printf("\n");fflush(stdout);
  }
}


main(int argc, char **argv){

  doub.e SOdb;
  doub.e startEbN0,endEbN0,stepEbN0;
  doub.e db,Eb,Pe;
  int rnamechar;
  int Numberofrun, count=0;
  doub.e *outPetable;
  int N_outPetable;

  char   *r1ifname, *r12fname;
  char paramfi.e[30];
  FILE *outPe,
  int i.


  if(argc == 1) {
    printf("usage: comp_pe_Bit paramfile\n");
    exit(0.,
  ,

  strcpy(paramfi.e,argv[1]);
  r1ifname = (char *)avread(argv[1],"r1ifname",&rnamechar,CHAR);
  r12fname = (char *)avread(argv[1],"r12fname",&rnamechar,CHAR);
  asread(paramfi.e,"Bd",&Bd,DOUBLE);
  asread(paramfi.e,"Ns",&Ns,INT);
  asread(paramfi.e,"Q",&Q,INT);
  asread(paramfi.e,"T0",&T0,DOUBLE);
  asread(paramfi.e,"S",&S,DOUBLE);
  asread(paramfi.e,"SOdb",&SOdb,DOUBLE);
  asread(paramfi.e,"startEbN0",&startEbN0,DOUBLE);
  asread(paramfi.e,"endEbN0",&endEbN0,DOUBLE);
  asread(paramfi.e,"stepEbN0",&stepEbN0,DOUBLE);
  asread(paramfi.e,"Numberofrun",&Numberofrun,INT);

  /*initia.ize abscissas and weights of the Gauss-Legendre quadrature formula*/
  gau_eg();.

  init_r1i(r1ifname);.
  init_r12(r12fname);.
  Eb = S;                          /**CHECK */
  S = S*pow(10,-SOdb/10);.
  init_bitmap();.

  sprintf(paramfi.e,"%sBdIgSOIgNsId",argv[1],Bd,SOdb,Ns);
  strcat(paramfi.e," bit out");.
  N_outPetab.e = (int)(endEbN0-startEbN0)/stepEbN0 + 1;
  outPetab.e = (doub.e*)ma...oc(sizeof(doub.e)*N_outPetable);
  for(i=0;i<N_outPetab.e;i++)
    outPetab.e[i]=0.
  do{
    setbitmap().
    for(db = startEbN0,i=0; db <= endEbN0; db += stepEbN0,i++) {
      N0 = Eb*pow(10,(-db/10)),
      Pe = comp_Pe();.
      outPetab.e[i] = outPetable[i]+ Pe;
      printf("%g %g\n",db,Pe);
    }
    count++;
    outPe = fopen(paramfile,"w");
    if(outPe==NULL){
      printf("Not able to open %s\n",paramfile);
      exit(0);
    }
    fprintf(outPe,"\"count=%d Bd%g Ns%d\"\n",count,Bd,Ns);
    for(i=0;i<N_outPetable;i++)
      fprintf(outPe,"%g %g\n",(startEbN0+i*stepEbN0),(outPetable[i]/count));
    fclose(outPe);
  }while(count<Numberofrun);
}
```

```
/*****************************************************
*
* Numerical integration (from Numerical Recipes)
*
*****************************************************/

void gauleg(void)
{

  int j,i;
  double z1, z, xl, pp, p3, p2, p1;

  xl = 1;

  for(i=0;i<GAULEG_N;i++){
    z = cos(M_PI*(i+0.75)/(2*GAULEG_N+0.5));
    do{
      p1=1.0;
      p2=0.0;
      for(j=0;j<(2*GAULEG_N);j++){
p3=p2;
p2=p1;
p1=((2.0*j+1)*z*p2-j*p3)/(j+1);
      }
      pp=2*GAULEG_N*(z*p1-p2)/(z*z-1);
      z1=z;
      z=z1-p1/pp;
    }while(fabs(z-z1)>EPS);
    x_pos[GAULEG_N-i-1]= xl*z;
    weg[GAULEG_N-i-1]=2.0*xl/((1.0-z*z)*pp*pp);
  }
}

double qgaus(double (*func)(double), double a, double b)
{
  int j;
  double xr, xm, dx, s;


  xm=0.5*(b+a);
  xr=0.5*(b-a);
  s=0;
  for(j=0;j<GAULEG_N;j++){
    dx=xr*x_pos[j];
    s += weg[j]*((*func)(xm+dx)+(*func)(xm-dx));
  }
  return s *=xr;
}




/*
local Variables:
compile-command "apcc -O -o comp_pe_Bit -g comp_pe_Bit.c -I/usr/include/.local -lpmatlib -llm -lm"
compile-command "gcc -o comp_pe_Bit -g comp_pe_Bit.c -I/usr/include/local -L/opt/lib -lpmatlib -llm -lm"
compile-command "gcc -o dcomp_pe_Bit -g comp_pe_Bit.c -I/usr/include/local -L/opt/lib -lpmatlib -llm -lm"
End
*/
```

Appendix C Programs Used to Evaluate the Matched Filter Bound

Algorithm 2 Compute data for matched filter bound

```
/*************************************
 *
 *  Program  comp_pe.c
 *
 *  Chet Lo
 *  Utah State University
 *
 *  Date   Oct 28, 1999
 *
 *************************************/
#include <pmat.lib.h>
#include <m.lib.h>
#include <assdtypes.h>
#include <stdio.h>
#include <std.lib.h>
#include <math.h>
#include <string.h>
#include <comp.x.h>
#include <grand.h>
#include <aruti..h>

#define min(a,b) ((a) < (b) ? (a) : (b))
#define max(a,b) ((a) < (b) ? (b) : (a))
#define calloc_array pcalloc_array
#define calloc_matrix pca..loc_matrix
#define DK_LEN 25000000
#define DK_prob_step 0.001
#define r2  1.41421356237

/* global variables */
int type;  /* 1: single function, 2: two functions. */
```

```
int find_ev;
double *auto_table, auto_unit, auto_step, auto_length;
int auto_no_terms;
double *cross_table1, cross_unit1, cross_step1, cross_length1;
int cross_no_terms1;
double *cross_table2, cross_unit2, cross_step2, cross_length2;
int cross_no_terms2;
int no_path;                    /* number of path */
double *c_tau, *alpha;          /*path delay, path root-mean-squ magnitude */
double **M, **N;                /* M the auto_corr matrix; N the cross_corr matrix */
double symbol_int_re;           /* relative value (unit) of the signal */
double symbol_interval;         /* absolute value (unit) of the signal */
dcomplx *Z;
double xMx, xNx;
double *dk0, *dk1;
double *dk0_prob, *dk1_prob;
double DK_prob_range;
int dk_prob_length;
double SN;
double integration_range;
/************************/
void init_autocorr_table(char *);          /*initiate auto_correlation table*/
void init_crosscorr_table(char *,char *);  /*initiate cross_correlation table*/
void init_channel(char *);                 /*initiate channel chara*/
double Rx(double);                         /*auto_corr with unity unit /sym_int */
double Rxy(double);                        /*cross_corr with unity unit / sym_int */
void init_M(void);
void init_N(void);
void init_Z(void);
void rand_Z(void);          /*generates complex Gaussian random N in Z */
double cal_xMx(void);
double cal_xNx(void);
void cal_dk0(void);
void ca._dk0_dk1(void);
void hpsort(unsigned long, double * );
double Q(double);
double qf(double);
void init_DK_prob(void);
double PE_dk0(double);
double PE_dk1(double);
double func_prob_dk0(double);
double func_prob_dk1(double);
void output(char *, char *);
/************************/

double simp.e(double (*func)(double), double, double);
double de.tat;
/**************************************/


main(int argc, char **argv){
  char *autocorrfname, *crosscorrfname1, *crosscorrfname2, *channelfname, *outputfname;
  int autocorrnamechar, crosscorrnamechar1, crosscorrnamechar2, channelnamechar, outputfnamechar;

  if(argc == 1) {
    printf("usage: comp_pe paramfilename\n");
    exit(0);
  }

  asread argv[1],"sym_int",&symbo._interval,DOUBLE);
  autocorr*name = (char *)avread(argv[1],"autocorr",&autocorrnamechar,CHAR);
  crosscorr*name1 = (char *)avread(argv[1],"crosscorr1",&crosscorrnamechar1,CHAR); /* + side */
  crosscorr*name2 = (char *)avread(argv[1],"crosscorr2",&crosscorrnamechar2,CHAR); /* - side */
  output*name = (char *)avread(argv[1],"outputfile",&outputfnamechar,CHAR);
  channe.fname = (char *)avread(argv[1],"channel",&channelnamechar,CHAR);
  asread argv[1],"De.te"",&de.tat,DOUBLE);          /* integration step */
  asread argv[1],"type",&type,INT);
  asread argv[1],"find_ev",&find_ev,INT);
  init_autocorr_tab.e(autocorrfname);
  if.type == 2
    symbo._int_re = 2*symbo._interval;
  e.se
    symbo._int_re = symbo._interval;
  if(type == 2)
    init_crosscorr_tab.e(crosscorrfname1,crosscorrfname2);
  init_channe.(channe.*name);
  init_M();
  if(type == 2)
    init_N();
  init_Z();

  if(type == 1 )
    ca._dk0();
  e.se
    ca._dk0_dk1();
  init_DK_prob();
  output(outputfname,channe.fname);
}

void init_autocorr_tab.e(char *filename)
{
  FILE *infile,
  double tau,
  int i,

  /* autocorr fi.e format:
     unit
     tota._length
     step_size
     no_of_terms
     tau   re.se
      .      .
      .      .
      .      .
     */

  infile = fopen(filename,"r");
```

```
  if(infile == NULL) {
    printf("Could not open autocorrfile\n");
    exit(0);
  }
  fscanf(infile,"%lf\n",&auto_unit);      /* basic unit length */
  fscanf(infile,"%lf\n",&auto_length);    /* # of unit the file contains */
  fscanf(infile,"%lf\n",&auto_step);      /* step size in unit between terms */
  fscanf(infile,"%d\n",&auto_no_terms);   /* # of terms */
  auto_table = (double *)calloc_array(DOUBLE,auto_no_terms,"auto_table");
  for(i=0;i<auto_no_terms;i++){
    fscanf(infile,"%lf%lf\n",&tau, &auto_table[i]);
  }
  fclose(infile);
}

void init_crosscorr_table(char *filename1, char *filename2)
{
  FILE *infile;
  double tau;
  int i;

  /* crosscorr file format:
     unit
     total_length
     step_size
     no_of_terms
     tau   value
      .      .
      .      .
      .      .
     */

  infile = fopen(filename1,"r");
  if(infile == NULL) {
    printf("Could not open crosscorr1 file\n");
    exit(0);
  }
  fscanf(infile,"%lf\n",&cross_unit1);
  fscanf(infile,"%lf\n",&cross_length1);
  fscanf(infile,"%lf\n",&cross_step1);
  fscanf(infile,"%d\n",&cross_no_terms1);
  cross_tab.c1 = (double *)calloc_array(DOUBLE,cross_no_terms1,"cross_table1");
  for(i=0;i<cross_no_terms1;i++){
    fscanf(infile,"%lf%lf\n",&tau, &cross_table1[i]);
  }
  fclose(infile);

  infile = fopen(filename2,"r");
  if(infile == NULL) {
    printf("Could not open crosscorr2 file\n");
    exit(0);
  }

  fscanf(infile,"%lf\n",&cross_unit2);
  fscanf(infile,"%lf\n",&cross_length2);
  fscanf(infile,"%lf\n",&cross_step2);
  fscanf(infile,"%d\n",&cross_no_terms2);
  cross_tab.c2 = (double *)calloc_array(DOUBLE,cross_no_terms2,"cross_table2");
  for(i=0;i<cross_no_terms2;i++){
    fscanf(infile,"%lf%lf\n",&tau, &cross_table2[i]);
  }
  fclose(infile);
}

void init_channe(char *channel)
{
  FILE *infile;
  int i;

  /* channe. file format
     no_path
     c_tau1   alpha1
     c_tau2   alpha2



     */

  infile = fopen(channe,"r");
  if(infile == NULL) {
    printf("Could not open channel file\n");
    exit(0);
  }
  fscanf(infile,"%d\n",&no_path);   /* number of pathes */
  c_tau = (double *)calloc_array(DOUBLE,no_path,"c_tau");
  alpha = (double *)calloc_array(DOUBLE,no_path,"alpha");
  for(i=0;i<no_path;i++)
    fscanf(infile,"%lf%lf\n",&c_tau[i], &alpha[i]);
  fclose(infile);
}

void init_Z(void){
  Z = (dcomplx *)malloc(sizeof(dcomplx)*no_path);
}


void rand_Z(void){
  int i;

  for(i=0;i<no_path;i++)
    Z[i] = dComplx((gran()/r2),(gran()/r2));
}


double Rx(double tau)
{
  double t, scaled_tau, delta;
```

```
  int n;

  tau /= symbol_int_re;
  if(tau>0)
    t = tau;
  else
    t = -tau;
  t -= auto_unit;
  scaled_tau = t / auto_step;
  n = (int) scaled_tau;
  if(n>=(auto_no_terms-1))
    return(auto_table[auto_no_terms-1]);
  else {
    delta = scaled_tau -n;
    return(auto_table[n] + delta*(auto_table[n+1]-auto_table[n]));
  }
}


double Rxy(double tau) /*Crosscorrelation*/
{
  double *cross_table, cross_unit, cross_step;
  int cross_no_terms;
  double t, scaled_tau, delta;
  int n;

  tau /= symbol_int_re;

  if(tau>0){
    t = tau;
    cross_table = cross_table1;
    cross_unit = cross_unit1;
    cross_step = cross_step1;
    cross_no_terms = cross_no_terms1;
  }
  else{
    t = -tau;
    cross_table = cross_table2;
    cross_unit = cross_unit2;
    cross_step = cross_step2;
    cross_no_terms = cross_no_terms2;
  }
  t -= cross_unit;
  scaled_tau = t / cross_step;
  n = (int) scaled_tau;
  if(n>=(cross_no_terms-1))
    return(cross_table[cross_no_terms-1]);
  else {
    delta = scaled_tau -n;
    return(cross_table[n] + delta*(cross_table[n+1]-cross_table[n]));
  }
}

void init_M(void)
{
  int i, j;
  double **Qm, **Lambda, *e_v, sum;

  M = (double **)calloc_matrix(DOUBLE,no_path,no_path,"M");
  for(i=0;i<no_path;i++)
    for(j=0;j<no_path;j++)
      M[i][j]=a.pha[i]*a.pha[j]*Rx((c_tau[i]-c_tau[j]));
  printf("M.\n");
  for(i=0;i<no_path;i++){
    for(j=0;j<no_path;j++)
      printf("%e ",M[i][j]);
    printf("\n");
  }
}


void init_M(void)
{
  int i, j;

  M = (double **)calloc_matrix(DOUBLE,no_path,no_path,"M");
  for(i=0;i<no_path;i++)
    for(j=0;j<no_path;j++)
      M[i][j]=a.pha[i]*a.pha[j]*Rxy((c_tau[i]-c_tau[j]));
}

double cal_xRx()
{
  int i, j;
  dcomplx temp, temp1;

  temp1 = dComp.x(0, 0);

  for(i=0;i<no_path;i++)
    for(j=0;j<no_path;j++){
      temp = dComplx(M[i][j], 0);
      temp = dCmul(temp, Z[i]);
      temp = dCmul(temp, dConjg(Z[j]));
      temp1 =  dCadd(temp, temp1);
    }
  return(temp1.re);
}

double cal_xHx()
{
  int i, j;
  dcomplx temp, temp1;


  temp1 = dComplx(0, 0);

  for(i=0;i<no_path;i++)
```

```
    for(j=0;j<no_path;j++){
      temp = dComplx(H[i][j], 0);
      temp = dCmul(temp, Z[i]);
      temp = dCmul(temp, dConjg(Z[j]));
      temp1 =  dCadd(temp, temp1);
    }
  return(temp1.re);
}

void cal_dk0_dk1()
{
  int i;
  double sqr_xMx, a, b ;

  dk0 = (double *)calloc_array(DOUBLE,(DK_LEN+1),"dk0");
  dk1 = (double *)calloc_array(DOUBLE,(DK_LEN+1),"dk1");

  for(i=0;i<=DK_LEN;i++){
    rand_Z();
    xMx = cal_xMx();
    xNx = cal_xNx();
    sqr_xMx = sqrt(xMx);
    dk0[i] = (xMx - xNx)/sqr_xMx;
    dk1[i] = (xMx + xNx)/sqr_xMx;
    /* Assume 1 is sent at this scale while k0: -1 on other scale
       k1: 1 on other scale
       */

  }

  hpsort(DK_LEN, dk0);
  hpsort(DK_LEN, dk1);
  for(i=0;i<DK_LEN;i++){
    dk0[i]=dk0[i+1];
    dk1[i]=dk1[i+1];
  }
  a = dk0[(DK_LEN-1)];
  b = dk1[(DK_LEN-1)];
  integration_range = max(a,b);
  DK_prob_range = integration_range;
}

void ca._dk0()
{
  int i.

  dk0 = (doub.e *)calloc_array(DOUBLE,(DK_LEN+1),"dk0");
  for(i=0,i<=DK_LEN;i++){
    rand_Z().
    xMx = ca._xMx();
    dk0[i] = sqrt(xMx);
  }
  hpsort(DK_LEN, dk0);
  for(i=0.i<DK_LEN;i++)
    dk0[i]=dk0[i+1];
  integration_range = dk0[(DK_LEN-1)];
  DK_prob_range = integration_range;
}

void init_DK_prob()
{
  int dk0_prob_i, dk1_prob_i, dk0_i,dk1_i, begin_i, end_i;
  doub e de.ta.

  dk_prob_.ength = (int)DK_prob_range/DK_prob_step;
  dk_prob_.ength += 1.
  dk0_prob = (doub.e *)ca..oc_array(DOUBLE,dk_prob_length,"dk0_prob");

  de.ta = DK_prob_step.
  dk0_i = 0.
  dk0_prob[0] = 0.
  for(dk0_prob_i=1.dk0_prob_i<=dk_prob_length;dk0_prob_i++,delta+=DK_prob_step){
    begin_i = dk0_i.
    whi.e(!dk0[dk0_i]<de.ta)&&(dk0_i<(DK_LEN-1)))
      dk0_i++.
    end_i = dk0_i.
    dk0_prob[dk0_prob_i] = (doub.e)(end_i - begin_i)/(DK_LEN*DK_prob_step);
  }
  pfree_array(dk0);

  if(type == 2)
    dk1_prob = (doub.e *)ca..oc_array(DOUBLE,dk_prob_length,"dk1_prob");
    de.ta = DK_prob_step.
    dk1_i = 0.
    dk1_prob[0] = 0.
    for(dk1_prob_i=1.dk1_prob_i<=dk_prob_length;dk1_prob_i++,delta+=DK_prob_step){
      begin_i = dk1_i;
      whi.e((dk1[dk1_i]<de.ta)&&(dk1_i<(DK_LEN-1)))
dk1_i++.
      end_i = dk1_i.
      dk1_prob[dk1_prob_i] = (double)(end_i - begin_i)/(DK_LEN*DK_prob_step);
    }
    pfree_array(dk1);
  }
}

double PE_dk0(double x)
{
  int a;
  double scaled_x;

  if(x==0)
    return(0);
  scaled_x = x/DK_prob_step;
  a = (int) scaled_x;

  if(a>=(dk_prob_length-1))
```

```
    return(0);
  else
    return(dk0_prob[n+1]);
}


double PE_dk1(double x)
{
  int n;
  double scaled_x;

  if(x==0)
    return(0);
  scaled_x = x/DK_prob_step;
  n = (int) scaled_x;

  if(n>=(dk_prob_length-1))
    return(0);
  else
    return(dk1_prob[n+1]);
}

double func_prob_dk0(double E)
{
  double N0, temp;

  temp = -(SN/10);
  N0 = pow(10,temp);
  temp = sqrt(N0);              /*Eb=1, Es=2 */
  temp = Q((E/temp))*PE_dk0(E);
  return(temp);
}


double func_prob_dk1(double E)
{
  double N0, temp;

  temp = -(SN/10);
  N0 = pow(10,temp);
  temp = sqrt(N0);              /*Eb=1, Es=2 */
  temp = Q((E/temp))*PE_dk1(E);
  return(temp);
}

void output(char *fname, char *cfname)
{
  FILE *fpt,
  char outputf1.e[30];
  double integra., average[500];
  int i,
  double tau,


  sprintf(outputf1.e,"%s_%d-%_teEs.perr",fname,type,symbol_interval,cfname);
  fpt = fopen(outputf1.e,"w");
  if(fpt == NULL) {
    printf("Cou.d not open output file! \n");
    exit(0);
  }

  i=0,
  for(SN=b,SN<2b,SN+=0.1){
    integra. = simp.e(func_prob_dk0,0,integration_range);
    if(type == 1){
      fprintf(fpt,"%f    %e\n",SN,integral);
      ff.ush(NULL),
    }
    average[i++] = integra.,
  }

  if(type==2){
    i = 0,
    for(SN=b,SN<2b,SN+=0.1){
      integra  = simp.e(func_prob_dk1,0,integration_range);
      average[i++] += integra.,
    }
    i = 0,
    for(SN=b,SN<2b,SN+=0.1){
      fprintf(fpt,"%f    %e\n",SN,(average[i++]/2));
    }
  }
  fc.ose(fpt),

  sprintf(outputf1.e,"%s %d-% teEs.pdf",fname,type,symbol_interval,cfname);
  fpt = fopen(outputf1.e,"w");
  if(fpt == NULL) {
    printf("Cou.d not open output file! \n");
    exit(0);
  }

  fprintf(fpt,"Title\ert: PDF Sym_interval:%f\n",symbol_interval);
  integral = simple(PE_dk0,0,integration_range);
  fprintf(fpt,"\"dk0 A:%f\"  \n",integral);
  for(tau=0;tau<=integration_range;tau += 0.1)
    fprintf(fpt,"%f  %f\n",tau,PE_dk0(tau));
  if(type==2){
    integral = simple(PE_dk1,0,integration_range);
    fprintf(fpt,"\n\"dk1  A:%f\"  \n",integral);
    for(tau=0;tau<=integration_range;tau += 0.1)
      fprintf(fpt,"%f  %f\n",tau,PE_dk1(tau));
  }
  fclose(fpt);
}
/*************************************************
*
* Numerical integration
```

```
*
**************************************************/

double simple(double (*func)(double), double a, double b)
{
  double tau,integral=0;

  for(tau=a;tau<=b;tau+=deltat)
    integral += (*func)(tau);

  integral *= deltat;
  return integral;
}

double Q(double z)
{
  double  a,b,c;

  b=z*z+5.51;
  a=0.85*sqrt(b)+1.65*z;
  c=(double)(-z*z/2);
  z=(double)exp(c)/a;
  return(z);
}

void hpsort(unsigned long n, double ra[])
{
  unsigned long i,ir,j,l;
  double rra;

  if (n < 2) return;
  l=(n >> 1)+1;
  ir=n;
  for (;;) {
    if (l > 1) {
      rra=ra[--l];
    } else {
      rra=ra[ir];
      ra[ir]=ra[1];
      if (--ir == 1) {
ra[1]=rra;
break;
      }
    }
    i=l;
    j=l<<1;
    while (j <= ir) {
      if (j < ir && ra[j] < ra[j+1]) j++;
      if (rra < ra[j]) {
ra[i]=ra[j];
i=j;
j <<= 1;
      } else j=ir+1;
    }
    ra[i]=rra;
  }
}
/* (C) Copr. 1986-92 Numerical Recipes Software #R4,1. */

/*
Local Variables:
compile-command "gcc -O -o comp_pe -g comp_pe.c -I/usr/include/local  -I/home/silksp/home1/include -L/opt/lib -lpmatlib -lim -lm -lcomplx"
compile-command "gcc -O -o comp_pe -g comp_pe.c -I/usr/include/local  -L/opt/lib -lpmatlib -lim -lm -lcomplx"
compile-command "gcc -O -o comp_pe -g comp_pe.c -I/usr/include/local -I/home/silksp/home1/include -L/opt/lib  -lpmatlib -lim -lm -lcomplx"
End
*/
```

References

[1] I. Daubechies, *Ten Lectures on Wavelets.* Philadelphia, PA: SIAM, 1992.

[2] T. K. Moon and C. Lo, "Wavelet multiscale signaling and its performance in a ricean fast fading channel," in *Proc. International Telemetering Conference*, pp. 345–354, Oct 27 1999.

[3] I. Daubechies, "Orthonormal Bases of Compactly Supported Wavelets," *Com. on Pure and Appl. Math.*, vol. XLI, pp. 909–996, 1988.

[4] M. Vetterli and J. Kovačević, *Wavelets and Subband Coding.* Prentice-Hall, 1995.

[5] C. K. Chui, *An Introduction to Wavelets.* Boston, MA: Academic Press, 1992.

[6] M. K. Simon, S. M. Hinedi, and W. C. Lindsey, *Digital Communication Techniques: Signal Design and Detection.* Englewood Cliffs, NJ: Prentice-Hall, 1995.

[7] A. R. Lindsey, *Generalized orthogonally multiplexed communication via wavelet packet bases.* PhD thesis, Ohio University, Athens, Ohio, June 1995.

[8] K. Murata and K. Hirade, "GMSK modulation of digital radio telephony," *IEEE Trans. Comm.*, vol. 29, pp. 1044–1050, Jul. 1981.

[9] L. J. Mason, "Error probability evaulation for systems employing differential detection in a Rician fast fading environment and Gaussian noise," *IEEE Trans. Comm.*, vol. 35, pp. 39–46, Jan 1987.

[10] R. F. Pawula, S. O. Rice, and J. H. Roberts, "Distribution of the phase angle between two vectors perturbed by Gaussian noise," *IEEE Trans. Comm.*, vol. 30, pp. 1828–1841, Aug 1982.

[11] F. Ling, "Matched Filter-Bound for Time-Discrete Multipath Rayleigh Fading Channels," *IEEE Trans. Comm.*, vol. 43, pp. 710–713, Feb/Mar/Apr 1995.

[12] K.-W. Yip and T.-S. Ng, "Matched Filter Bound for Multipath Rician-Fading Channels," *IEEE Trans. Comm.*, vol. 46, pp. 441–445, Apr 1998.

[13] J. Mazo, "Exact Matched Filter Bound For Two-Bean Rayleigh Fading," *IEEE Trans. Comm.*, vol. 39, pp. 1027 1030, Jul 1991.

[14] B. Sklar, "Rayleigh Fading Channels in Mobile Digital Communication Systems, Part I: Characterization," *IEEE Commun. Mag.*, vol. 35, pp. 136–146, Sep 1997.

[15] R. E. Blahut, *Digital Transmission of Information.* Reading, MA: Addison Wesley, 1990.

[16] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* New York: Wiley, 1991.

[17] R. G. Gallager, *Information Theory and Reliable Communication.* New York: Wiley, 1968.

[18] K. K. Noru, "Adapting wavelets to match desired signal spectrum," Master's thesis, Utah State University, 1997.

[19] D. Rappaport, "Constant directions of the riccati equation," *Automatica*, vol. 8, pp. 175–186, 1972.

[20] A. F. Naguib, N. Seshadri, and A. R. Calderbak, "Increasing data rate over wireless channels," *IEEE Sig. Proc. Mag.*, vol. 17, pp. 76–92, May 2000.

[21] B. Sklar, "Rayleigh Fading Channels in Mobile Digital Communication Systems, Part II: Mitigation," *IEEE Commun. Mag.*, vol. 35, pp. 148–155, Sep 1997.

[22] K. Brayer, "HF Data Transmission: Lessons from the Past, Directions for the Future," *IEEE Journal on Selected Areas in Communications*, vol. SAC-5, pp. 90–101, Feb 1987.

[23] R. Price, "The detection of signals perturbed by scatter and noise," *IRE Trans. Inform. Theory*, vol. PGIT-4, pp. 163–170, Sep 1954.

[24] R. Price, "Optimum detection of random signals in noise, with application to scatter-multipath communication," *IRE Trans. Inform. Theory*, vol. IT-2, pp. 125–135, Dec 1956.

[25] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering*. New York: Wiley, 1965.

[26] P. A. Bello, "Characterization of randomly time-variant linear channels," *ieeecom*, pp. 360–393, December 1963.

[27] R. S. Kennedy, *Fading Dispersive Communication Channels*. Wiley-Interscience, 1969.

[28] K. Brayer, ed., *Data Communications via Fading Channels*. IEEE Press, 1975.

[29] S. Stein, "Fading Channel Issues in System Engineering," *IEEE Journal on Selected Areas in Communications*, vol. SAC-5, pp. 68–89, Feb 1987.

[30] J. G. Proakis, *Digital Communications, 3rd ed.* New York, NY: McGraw Hill, 1995.

[31] E. Biglieri and M. Elia, "Multidimensional Modulation and Coding for Band-limited Digital Channels," *IEEE Trans. IT*, vol. 34, no. 4, pp. 803–809, July 1988.

[32] G. W. Wornell and A. V. Oppenheim, "Wavelet-based Representations for a Class of Self-similar Signals with Applications to Fractal Modulation," *IEEE Trans. IT*, vol. 38, pp. 785–800, March 1992.

[33] P. Bello, "A Troposcatter Channel Model," *IEEE Trans. Comm. Tech.*, vol. 17, pp. 130–137, Apr. 1969.

[34] D. R. Hummels and R. W. Ratcliffe, "Calculation of error probability for msk and oqpsk systems operating in a fading multipath environment," *IEEE Trans. Veh. Technol.*, vol. VT-30, pp. 112–120, Aug 1981.

[35] P. Bello, "Aeronautical channel characterization," *IEEE Trans. Comm.*, vol. COM-21, pp. 548–563, May 1962.

[36] W. C. Jakes, ed., *Microwave Mobile Communications*. New Youk: Wiley, 1974.

[37] P. Bello and B. Nelin, "The influence of fading spectrum on the binary error probabilities of incoherent and differentially coherent matched filter receivers," *IRE Trnas. Commun. Sys.*, vol. CS-10, pp. 160–168, June 1962.

[38] A. B. Glenn and G. Lieberman, "Performance of digital communications systems in an arbitrary fading rate and jamming environments," *IEEE Trans. Comm. Sys.*, vol. CS-11, pp. 57–68, Mar 1963.

[39] G. F. Montgomery, "Message error in diversity frequency-shift reception," *Proceddings of the IRE*, pp. 1184 1187, Jul 1954.

[40] J. Pierce, "Theoretical diversity improvement in frequency-shift keying," *Proceddings of the IRE*, pp. 903 910, May 1958.

[41] D. Brennan, "Linear diversity combining techniques," *Proceddings of the IRE*, pp. 1075–1102, Jun 1959.

[42] J. Pierce, "Theoretical limitations of frequency and time diversity for fading binary transmissions," *IRE Trnas. Commun. Sys.*, pp. 186–189, Jun 1961.

[43] G. L. Turin, "On optimal diversity reception," *IRE Trnas. Infor. Th.*, pp. 154–166, Jul 1961.

[44] M. V. Clark, L. J. Greenstein, W. K. Kennedy, and M. Shafi, "Matched filter performance bounds for diversity combining receivers in digital mobile radio," *IEEE Trans. Veh. Technol.*, vol. 41, pp. 356–362, Nov 1992.

[45] D. Bouras, P. Mathiopoulos, and D. Makrakis, "Optimal detection of coded differentially encoded QAM and PSK signals with diversity reception on correlated fast Ricean fading channels," *IEEE Trans. Veh. Technol.*, vol. 42, pp. 245–258, Aug. 1993.

[46] J. Ventura-Traveset, G. Caire, E. Biglieri, and G. Taricco, "Impact of diversity reception on fading channels with coded modulation - part i: Coherent detection," *IEEE Trans. Comm.*, vol. 45, pp. 563–572, May 1997.

[47] A. J. Viterbi and I. M. Jacobs, "Advances in coding and modulation for noncoherent channels affected by fading, partial band, and multiple-access interference," in *Advances in Communication Systems* (A. J. Viterbi, ed.), New York: Academic, 1975.

[48] D. Chase, "Digital signal design concepts for a time-varying ricean channel," *IEEE Trans. Comm.*, vol. COM-24, pp. 164–172, Feb 1976.

[49] J. N. Pieper, J. G. Proakis, R. R. Reed, and J. K. Wolf, "Design of efficient coding and modulation for a rayleigh fading channel," *IEEE Trans. Infor. Th.*, vol. IT-24, pp. 457–468, Jul 1978.

[50] I. Rahman, *Bandwidth Constrained Signal Design for Digital Communication over Rayleigh Fading Channels and Partial Band Interference Channels.* PhD thesis, Northeastern University, Boston, Mass., 1981.

[51] G. W. Wornell, "Spread-response precoding for communication over fading channels," *IEEE Trans. Infor. Th.*, vol. 42, pp. 488–501, Mar 1996.

[52] D. Rainish, "Diversity transform for fading channels," *IEEE Trans. Comm.*, vol. 44, pp. 1653–1661, Dec 1996.

[53] A. M. Sayeed and B. Aazhang, "Joint multipaht-doppler diversity in mobile wireless communications," *IEEE Trans. Comm.*, vol. 47, pp. 123–132, Jan 1999.

[54] J. Ventura-Traveset, G. Caire, E. Biglieri, and G. Taricco, "Impact of diversity reception on fading channels with coded modulation - part iii: Co-channel interference," *IEEE Trans. Comm.*, vol. 45, pp. 809 818, Jul 1997.

[55] B. M. Hochwald and T. L. Marzetta, "Unitary space-time modulation for multiple-antenna communications in rayleigh flat fading," *IEEE Trans. Infor. Th.*, vol. 46, pp. 543–564, Mar 2000.

[56] V. M. DaSilva and E. S. Sousa, "Fading-resistant modulation using several transmitter antennas," *IEEE Trans. Comm.*, vol. 45, pp. 1236–1244, Oct 1997.

[57] T. K. Y. Lo, "Maximum ratio transmission," *IEEE Trans. Comm.*, vol. 47, pp. 1458–1461, Oct 1999.

[58] G. M. Vitetta, U. Mengali, and D. P. Taylor, "Double-filter differential detection of psk signals transmitted over linearly time-selective rayleigh fading channels," *IEEE Trans. Comm.*, vol. 47, pp. 239–247, Feb 1999.

[59] M. P. C. Fossorier and S. Lin, "Soft decision decoding of linear block codes based on ordered statistics for the rayleigh fading channel with coherent detection," *IEEE Trans. Comm.*, vol. 45, pp. 12–14, Jan 1997.

[60] M. Visintin, "Differential psk block demodulation over a flat correlated rayleigh-fading channel," *IEEE Trans. Comm.*, vol. 45, pp. 9–11, Jan 1997.

[61] F. Patenaude, J. H. Lodge, and Y.-Y. Chouinard, "Noncoherent diversity reception over nakagami-fading channels," *IEEE Trans. Comm.*, vol. 46, pp. 985–991, Aug 1998.

[62] I. B. C. Linda M. Davis and R. J. Evans, "Coupled estimatiors for equalization of fast-fading mobile channels," *IEEE Trans. Comm.*, vol. 46, pp. 1262–1265, Oct 1998.

[63] H.-N. Lee and G. J. Pottie, "Fast adaptive equalization/diversity combining for time-varying dispersive channels," *IEEE Trans. Comm.*, vol. 46, pp. 1146–1162, Sep 1998.

[64] S. D. Fina and G. E. Corraza, "Bayesian approach for erasure insertion in frequency-hop multiple-access communications with selective fading," *IEEE Trans. Comm.*, vol. 48, pp. 282–289, Feb 2000.

[65] R. Cusani and J. Mattila, "Equalization of digital radio channels with large multipath delay for cellular land mobile applications," *IEEE Trans. Comm.*, vol. 47, pp. 348–351, Mar 1999.

[66] D. K. Borah and B. D. Hart, "A robust receiver structure for time-varying, frequency-flat, rayleigh fading channels," *IEEE Trans. Comm.*, vol. 47, pp. 360–364, Mar 1999.

[67] W. S. Leon, U. Mengali, and D. P. Taylor, "Equalization of linearly frequency-selective fading channels," *IEEE Trans. Comm.*, vol. 45, pp. 1501–1503, Dec 1997.

[68] W. S. Leon and D. P. Taylor, "An adaptive receiver for the time- and frequency-selective fading channel," *IEEE Trans. Comm.*, vol. 45, pp. 1548–1555, Dec 1997.

[69] Y. A. Chau and J.-K. Wang, "Spectral-estimation-based acquisition for frequency-hopping spread-spectrum communications in a nonfading or rayleigh-fading channel," *IEEE Trans. Comm.*, vol. 45, pp. 445–455, Apr 1997.

[70] A. Narula, M. J. Lopez, M. D. Trott, and G. W. Wornell, "Efficient use of side information in multiple-antenna data transmission over fading channels," *IEEE Trans. Comm.*, vol. 16, pp. 1423–1436, Oct 1998.

[71] S. Bhashyam, A. M. Sayeed, and B. Aazhang, "Time-selective signaling and reception for communication over multipath fading channels," *IEEE Trans. Comm.*, vol. 48, pp. 83–94, Jan 2000.

[72] A. J. Goldsmith and S.-G. Chua, "Adaptive coded modulation for fading channels," *IEEE Trans. Comm.*, vol. 46, pp. 595–602, May 1998.

[73] T. Muller and H. Rohling, "Channel coding for narrow-band rayleigh fading with robustness against changes in doppler spread," *IEEE Trans. Comm.*, vol. 45, pp. 148–151, Feb 1997.

[74] T. K. Moon and W. C. Stirling, *Mathematical Methods and Algorithms for Signal Processing.* Upper Saddle River, New Jersey: Prentice Hall, 2000.

[75] F. G. Tricomi, *Integral Equations.* London and New York: Interscience Publishers, 1957.

[76] U. Hansson and T. M. Aulin, "Aspects on single symbol fading on the frequency flat Rayleigh fading channel," *IEEE Trans. Comm.*, vol. 47, pp. 874–883, June 1999.